UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Utilização de Técnicas de Redução de Dimensionalidade em Algoritmos de Otimização com Muitos Objetivos no Problema de Sincronização de Semáforos

Dissertação de Mestrado

Jonatas Cezar Vieira Santos

São Cristóvão – Sergipe

2019

UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Jonatas Cezar Vieira Santos

Utilização de Técnicas de Redução de Dimensionalidade em Algoritmos de Otimização com Muitos Objetivos no Problema de Sincronização de Semáforos

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de mestre em Ciência da Computação.

Orientador(a): Prof. Dr. André Britto de Carvalho
Coorientador(a): Prof. Dr. Leonardo Nogueira Matos

São Cristóvão – Sergipe

2019

Jonatas Cezar Vieira Santos

Utilização de Técnicas de Redução de Dimensionalidade em Algoritmos de Otimização com Muitos Objetivos no Problema de Sincronização de Semáforos/ Jonatas Cezar Vieira Santos. – São Cristóvão – Sergipe, 2019-

64 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. André Britto de Carvalho

Dissertação de Mestrado – UNIVERSIDADE FEDERAL DE SERGIPE
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO, 2019.

1. Sistemas de Transporte Inteligente. 2. Sincronização de Semáforos. 3. Otimização com Muitos Objetivos. 4. Redução de Objetivos. I. Orientador Prof. Dr. André Britto de Carvalho. II. Universidade Federal de Sergipe. III. Programa de Pós-graduação em Ciências da Computação. IV. Utilização de Técnicas de Redução de Dimensionalidade em Algoritmos de Otimização com Muitos Objetivos no Problema de Sincronização de Semáforos

CDU 02:141:005.7

Dedico esta dissertação à toda minha família, amigos e professores que me deram o apoio necessário para chegar aqui.

Resumo

A mobilidade urbana é um problema atual da sociedade moderna e dos grandes centros urbanos. Os Sistemas Inteligentes de Transporte (ITS) utilizam a tecnologia para tentar resolver esses problemas de mobilidade. No contexto de ITS, a gestão de tráfego é uma área que utiliza novos conceitos de organização e manutenção do tráfego, buscando obter um fluxo de tráfego de qualidade. A sincronização de semáforos é um deles e seu principal objetivo é garantir que os veículos tenham uma boa fluidez no trânsito, garantindo percorrer um trajeto em menos tempo possível. Com uma sincronização atingida, as medidas de qualidade tendem a melhorar, como a redução de emissão de poluentes, consumo de combustível, tempo de atraso, velocidade média global e outras. Indicar o melhor tempo semafórico é uma tarefa bastante complexa. É difícil modelar uma situação real, pois existem cadeias de cruzamentos, com características diferentes. A otimização em sincronização de semáforos se classifica como problema NP-Completo, a dificuldade do problema cresce exponencialmente, quando os números de variáveis de decisão e de medidas de qualidades aumentam. Sendo assim, nenhuma técnica clássica seria capaz resolvê-lo em um tempo razoável. Uma solução, é modelar o problema como de otimização, através de um simulador de tráfego. Com o simulador é capaz de construir uma representação computacional semafórica, composta por vias, rotas, veículos, cruzamentos e semáforos. A partir de configurações de condições de fluxo em cenários diferentes, podemos obter essas medidas de qualidades, tratadas como objetivos, extraídas do próprio simulador. O problema é modelado como de otimização multiobjetivo e por trabalhar com mais de 3 funções objetivos, é classificado como de otimização com muitos objetivos. Algoritmos tradicionais enfrentam problemas na otimização com muitos objetivos, uma das técnicas para resolver é a redução de objetivos. O objetivo desse trabalho é utilizar técnicas de aprendizagem de máquina de redução de dimensionalidade, para redução de objetivos no problema de sincronização de semáforos. Foram aplicadas duas técnicas na busca de identificar os objetivos essenciais e descartar os demais para reduzir. As técnicas trabalhadas foram o L-PCA e o K-PCA utilizando os kernels polinomial, RBF e sigmoide. Uma otimização, utilizando os algoritmos NSGA-II e NSGA-III, foi aplicada nos conjuntos contendo todos os objetivos, foram trabalhados 12, e também para os subconjuntos obtidos pela redução. Foram feitas comparações das otimizações entre os conjuntos sem redução e os subconjuntos reduzidos. Além do mais, foram executados testes para identificar se houve diferença estatística entre os algoritmos. Os resultados mostraram que o NSGA-III obteve melhores resultados, e o K-PCA com kernel polinomial foi o melhor algoritmo de redução, conseguindo até superar o NSGA-III sem redução. Concluiu-se também que não houve diferença estatística entre os algoritmos. Portanto, trabalhar com um conjunto menor de objetivos, se tem um desempenho melhor na otimização sem perder a qualidade das informações.

Palavras-chave: Sistemas de Transporte Inteligente, Sincronização de Semáforos, Otimização com Muitos Objetivos, Redução de Objetivos.

Abstract

Urban mobility is a current problem of modern society and large urban centers. Intelligent Transportation Systems (ITS) use technology to address these mobility issues. In the context of ITS, traffic management is an area that uses new concepts of organization and maintenance of traffic, seeking to achieve a flow of quality traffic. The traffic light synchronization is one of them and its main objective is to ensure that the vehicles have a good fluidity in the traffic, ensuring to cross a route in the shortest possible time. With timing achieved, quality measures tend to improve, such as reducing pollutant emissions, fuel consumption, delay time, overall average speed, and so on. Indicating the best traffic light is a very complex task. It is difficult to model a real situation because there are chains of crosses with different characteristics. The optimization in semaphore synchronization classifies as NP-Complete problem, the difficulty of the problem grows exponentially, when the numbers of decision variables and measures of qualities increase. Therefore, no classical technique would be able to solve it in a reasonable time. One solution is to model the problem as of optimization, through a traffic simulator. With the simulator, it is able to construct a traffic signal representation, composed of roads, routes, vehicles, intersections and traffic lights. From configurations of flow conditions in different scenarios, we can obtain these measures of qualities, treated as objectives, extracted from the simulator itself. The problem is modeled as multiobjective optimization and by working with more than 3 objective functions, it is classified as optimization with many objectives. Traditional algorithms face problems in optimization with many goals, one of the techniques to solve is the reduction of goals. The objective of this work is to use dimensionality reduction machine learning techniques to reduce objectives in the problem of synchronization of traffic lights. Two techniques were applied in the search to identify the essential objectives and discard the others to reduce. The techniques studied were L-PCA and K-PCA using the polynomial, RBF and sigmoid kernels. An optimization, using the NSGA-II and NSGA-III algorithms, was applied in the sets containing all the objectives, were worked 12, and also for the subsets obtained by the reduction. Comparisons of the optimizations between the full sets and the reduced subsets were made. We also performed tests to identify if there was statistical difference between the algorithms. The results showed that the NSGA-III obtained better results, and the K-PCA with polynomial kernel was the best reduction algorithm, even managing to overcome NSGA-III without reduction. It also concluded that there was no statistical difference between the algorithms, thus, to work with a smaller set of objectives, if it has a better performance in the optimization without losing the information quality.

Keywords: Intelligent Transport Systems, Traffic Synchronization, Many Objective Optimization, Goal Reduction.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ITS	Sistemas Inteligentes de Transporte
NSGA-II	Non-dominated Sorting Genetic Algorithm-II
NSGA-III	Non-dominated Sorting Genetic Algorithm-III
TIC	Tecnologia da Informação e Comunicação
SUMO	Simulation of Urban Mobility
MaOPs	Many-Objective Optimization Problems (Problemas de Otimização com Muitos Objetivos)
MOEA	Multi-Objective Evolutionary Optimization Algorithms (Algoritmos Evolucionários Multiobjetivo)
MOPs	Multi-Objective Optimization Problems (Problemas de Otimização Multi-Objetivo)
MOPSO	Multiobjective Particle Swarm Optimization (Otimização Multiobjetivo por Enxame de Partículas)
EAs	Evolutionary Algorithms (Algoritmos Evolucionários)
PCA	Análise de Componentes Principais
L-PCA	Linear - Análise de Componentes Principais
NL-PCA	Nonlinear - Análise de Componentes Principais
RBF	Radial Basis Function

Sumário

1

Introdução

Com o aumento do número de veículos em área urbana e a falta de espaço para construção de novos acessos, os problemas de mobilidade urbana vêm se agravando. Assim, é necessário o uso de mecanismos eficientes para o controle de tráfego. Atualmente, encontramos faixas exclusivas, mudanças nos sentidos das vias, viadutos, passarelas, túneis e novos semáforos usados como solução (ABDOOS; MOZAYANI; BAZZAN, 2011; MCKENNEY; WHITE, 2013).

Uma das soluções adotadas no controle de tráfego é a sincronização de semáforos, que é um sistema que agrega ao menos dois semáforos executados em conjunto, que façam com que um veículo percorra um trajeto sem realizar paradas. Indicar qual o melhor tempo semafórico em diferentes horários cenários é uma tarefa bastante complexa. É difícil modelar uma situação real, pois existem cadeias de cruzamentos com semáforos, com características diferentes. A dificuldade do problema cresce exponencialmente, devido ao grande número de variáveis de decisão e de medidas de qualidades que podem ser avaliadas. Esses problemas são classificados como NP-Completo e nenhuma técnica clássica ou manual é capaz resolvê-lo em um tempo razoável (OLIVEIRA; BAZZAN, 2006; MATOS, 2017).

A sincronização de semáforos pode ser resolvida como um problema de Otimização. A otimização, é uma técnica computacional para a resolução de problemas de maneira automática gerando possíveis melhores soluções. Para modelar o problema como de Otimização, é necessário o uso de um simulador de tráfego, sendo possível construir uma simulação hipotética de um cenário real, onde dada configurações de condições de fluxo em cenários diferentes, será capaz obter medidas de qualidades resultantes do próprio simulador (KHAMIS; GOMAA, 2014). Essas medidas são modeladas como os objetivos que serão otimizados. Serão otimizadas em busca de tempos ideais para a melhoria dessas medidas, como por exemplo, tempo de viagem, número de veículos que chegam ao seu destino, velocidade média global e fluxo de tráfego, emissão de dióxido de carbono (KHAMIS; GOMAA, 2014; MATOS, 2017). Para otimização dessas

medidas, é representado computacionalmente as possíveis combinações de tempo em forma de um vetor, onde cada posição representa o tempo "de verde", de cada fase dos semáforos definidos na simulação (CARVALHO, 2013; SUN; BENEKOHAL; WALLER, 2003).

Como existe um grande número de medidas, o problema pode ser classificado como um Problema de Otimização com Muitos Objetivos. Essa classe de problemas refere-se aos problemas de Otimização Multiobjetivo que possuem mais de 3 funções a serem otimizadas (KHAMIS; GOMAA, 2014).

Atualmente existe uma série de trabalhos na literatura que utilizam algoritmos de otimização aplicados à sincronização de semáforos. (SHEN; WANG; WANG, 2013) e (KWASNICKA; STANEK, 2006) trabalharam com sincronização de semáforos, ambos utilizando algoritmos genéticos para a otimização, e tentam otimizar duas medidas de qualidades. A maioria dos trabalhos encontrados, buscam otimizar até no máximo três medidas de qualidades, já Matos (2017) trabalha com seis medidas de qualidades (tempo de viagem e paragem, número de veículos que chegam ao seu destino, velocidade média global e fluxo de tráfego, emissão de dióxido de carbono). Na pesquisa foi construído um sistema com o uso de um simulador de trânsito e a execução de dois algoritmos evolucionários, o NSGA-II e o NSGA-III para a otimização das medidas de qualidades.

A redução de objetivos é um caminho interessante para trabalhar com otimização de muitos objetivos. São técnicas que busca identificar os objetivos conflitantes, preservar os objetivos essenciais, que possuem a maioria das informações do problema, e descartar os demais, chegando a uma redução(SAXENA et al., 2013; BROCKHOFF, 2006). Em Matos (2017) também explorou a redução de objetivos, trabalhou uma técnica proposta por (BROCKHOFF, 2006), que visava identificar e reduzir as medidas com alguma correlação entre si. Com isso, re-executar os algoritmos com um número menor de objetivos e obter uma melhora dos resultados considerando todas as seis medidas de qualidade. Dessa forma, atingiu bons resultados e indicou ser um bom caminho para a resolução do problema de sincronização de semáforos com muitos objetivos.

Apesar de obter bons resultados, o trabalho proposto por (MATOS, 2017) ainda pode ser melhorado em alguns pontos. Foi explorada apenas uma técnica de redução de objetivos, no entanto, existem outras técnicas na literatura, que poderiam ser aplicadas no problema. E também quanto ao pequeno número de objetivos trabalhados, pois não utilizou todas as medidas de qualidades calculadas pelo simulador.

1.1 Objetivos

Este trabalho tem como objetivo geral investigar técnicas de redução de dimensionalidade para a resolução do problema de otimização com muitos objetivos da sincronização de semáforos. A essência do estudo será a exploração de técnicas de redução de objetivos que apresentam bons resultados no contexto da aprendizagem de máquina que ainda não foram exploradas na

otimização. Será explorado nesse trabalho, um *framework* de redução de objetivos, que explora técnicas de redução de dimensionalidade, proposto por Saxena et al. (2013). Por fim, comparar os resultados e identificar se houve melhora na sincronização de semáforos.

1.1.1 Objetivos específicos

- Aplicar técnicas de redução de objetivos da literatura no problema de sincronização de semáforos;
- Aplicar técnicas de redução de objetivos ainda não exploradas no contexto de Otimização com Muitos Objetivos.
- Executar um conjunto de experimentos para avaliar os algoritmos propostos no problema da sincronização de semáforos.

A modelagem do problema de semáforos como de Muitos Objetivos é nova e pouco explora da na literatura. Além disso, poucos trabalhos estudam técnicas de aprendizagem de máquina na Otimização com Muitos Objetivos. Até então, não há estudo que usem essas técnicas nos problemas de sincronização de semáforos, então iremos aplicar essas técnicas no problema citado e comparar os resultados.

É conhecido na literatura que a os algoritmos evolucionários multiobjetivo apresentam melhores resultados quando o número de funções é pequeno. Assim, identificando um conjunto menor de funções a serem otimizadas, possibilita a exploração de mais algoritmos de otimização. Identificar um conjunto de medidas mais significantes auxilia o tomador de decisão. O tomando de decisão precisa analisar apenas as medidas identificadas como mais relevantes.

Como o modelo proposto por (SAXENA et al., 2013) obteve bons resultados, planejamos reproduzir e modificar a técnica descrita para aplicar no problema de sincronização de semáforos. A técnica pretende reduzir os objetivos do problema com Muitos Objetivos, e com isso podemos alcançar um melhor desempenho na solução dos problemas de sincronização de semáforos. Este receberá como entrada os dados iniciais, que passarão por todo o processo automático (execução de algoritmos, simulação, redução de objetivos e reexecução de algoritmos), e como saída terá as melhores soluções para o problema.

Para atingir os objetivos, a seguinte metodologia foi seguida. Inicialmente foi configurado o fluxo e cenário no simulador de trânsito. Como entrada, foi utilizada uma base de dados, que trabalha 12 funções objetivos, que são, *Depart delay, Trip duration, Wait steps, Time loss, CO abs, CO_2 abs, Fuel abs, HC abs, PM_x abs, NO_x abs, Global mean speed e Idle time*. Com a base de dados, duas técnicas de redução de objetivos foram aplicadas, uma linear, denominada L-PCA, e outra não linear, chamada K-PCA, que ainda utilizou uma variação de três kernels, polinomial, RBF e sigmoide. Essas técnicas, buscam identificar os objetivos essenciais e desprezar os demais, afim de obter como saída subconjuntos menores de objetivos. A partir desse ponto, as medidas de

qualidades são tratadas como objetivos. Foi feita uma otimização, usando os algoritmos NSGA-II e NSGA-III a partir dos dados sem redução, e dos subconjuntos reduzidos. A ideia é que não se precise otimizar todas as medidas de qualidade, e sim trabalhar com uma menor quantidade de objetivos, e poder chegar a uma otimização de todas as medidas envolvidas no problema. Comparações serão feitas entre todas as execuções de otimização, tanto pra os dados reduzidos, quanto para os dados sem redução. Também serão feitos testes para identificar se existe diferença estatística entre as técnicas, não havendo diferença, pode-se afirmar que é melhor trabalhar com conjuntos menores de objetivos e obter a mesma qualidade de resultados.

O desenvolvimento inicial deste trabalho gerou uma publicação na conferência IEEE Symposium on Computers and Communications (VIEIRA et al., 2018)

1.2 Estrutura do Documento

Este documento está organizado por capítulos, que são:

- Capítulo 2 - Otimização com Muitos Objetivos: apresenta os principais conceitos a respeito da otimização multiobjetivo e com muitos objetivos, e juntamente com dois algoritmos bastante utilizados na literatura;
- Capítulo 3 - Sincronização de Semáforos: apresenta os conceitos principais relacionados a cidades inteligentes, gestão de tráfego focando na sicronização de semáforos, as tecnologias utilizadas e simuladores de tráfego;
- Capítulo 4 - Técnicas de Redução de Objetivos no Problema de Sincronização de Semáforos com Muitos Objetivos: Apresenta o que foi proposto para a resolução do problema, incluindo a descrição das técnicas exploradas;
- Capítulo 5 - Experimentos: descreve o *benchmark* utilizado nos experimentos, detalhes do simulador utilizado, define os parâmetros dos algoritmos, e os resultados dos experimentos;
- Capítulo 6 - Conclusão: são apresentadas as conclusões e os trabalhos futuros.

2

Otimização Multiobjetivo e com Muitos Objetivos

Este capítulo tem o principal foco principal apresentar as definições de Otimização Multiobjetivo e de Otimização com Muitos Objetivos. O capítulo está organizado da seguinte forma: a seção 2.1 apresenta as definições de otimização multiobjetivo; na Seção 2.2 é discutido os conceitos os conceitos de otimização com mais de 3 objetivos, ou seja, com muitos objetivos; na Seção 2.3 são mostrados princípios da Redução de Objetivos; seção 2.4 são apresentados os algoritmos multiobjetivo NSGA-II e NSGA-III.

2.1 Conceitos de Otimização Multiobjetivo

A Otimização Multiobjetivo tem a particularidade de resolver problemas que pretendem otimizar de forma simultânea duas ou mais funções objetivo. Os problemas de Otimização Multiobjetivo (MOP, do inglês *Multi-Objective Optimization Problems*) tem uma característica de possuir mais de uma função objetivo em conflito, que se explica da seguinte forma: quando se atinge uma melhoria em uma função objetivo, os valores das outras funções tendem a piorar. Com essa situação, não se aplica encontrar apenas uma melhor solução para o problema, mas sim um conjunto de melhores soluções que mais sejam aceitáveis(CARVALHO, 2013). A seguir serão expressas algumas definições adaptadas de (COELLO et al., 2007; CARVALHO, 2013):

Nos problemas de otimização multiobjetivo, a solução é modelada em forma de um vetor de variáveis de decisão. Um problema de otimização multiobjetivo possui várias funções a serem otimizadas(duas ou mais), denominadas funções objetivo, essas são representadas por $f_j(\overrightarrow{x}), F : R^n \rightarrow R$, onde $j = \{2, ..., m\}$ é índice da função objetivo e m é o número de funções do problema. Estas funções são inseridas em um vetor e esse passa a ser apresentado por $\overrightarrow{f(\overrightarrow{x})} \in \Lambda$, no qual Λ é o espaço do vetor das funções objetivo para o problema exposto.

A seguir, é definida a representação de otimização multiobjetivo:

$$Minimiza \overrightarrow{f(\overrightarrow{x})} = (f_1(\overrightarrow{x}), f_2(\overrightarrow{x})..., f_m(\overrightarrow{x})) \tag{2.1}$$

no qual m representa o número de funções objetivo e a dimensão de Λ. Dada uma solução $\overrightarrow{x}$ no espaço de variáveis de decisão é mapeada para o vetor $\overrightarrow{f(\overrightarrow{x})}$ no espaço de funções objetivo, $F : R^n \rightarrow \Lambda$. A otimização multiobjetivo é definida pela Teoria da Otimizalidade de Pareto. As principais definições são:: Dominância de Pareto; Pareto Ótimo; Conjunto Pareto Ótimo; Fronteira de Pareto; Vetor Ideal e Vetores Extremos.

Dominância de Pareto: Dado $\overrightarrow{f(\overrightarrow{x})} = (f_1(\overrightarrow{x}), f_2(\overrightarrow{x})..., f_m(\overrightarrow{x}))$ e $\overrightarrow{f(\overrightarrow{y})} = (f_1(\overrightarrow{y}), f_2(\overrightarrow{y})..., f_m(\overrightarrow{y}))$, $\overrightarrow{f(\overrightarrow{x})}$ ***domina*** $\overrightarrow{f(\overrightarrow{y})}$, é apresentado por $\overrightarrow{f(\overrightarrow{x})} \preceq \overrightarrow{f(\overrightarrow{y})}$, se somente se (minimização): $\forall i \in \{1,2,...,m\} : f_i(\overrightarrow{x}) \leq f_i(\overrightarrow{y})$, e $\exists i \in \{1,2,...,m\} : f_i(\overrightarrow{x}) < f_i(\overrightarrow{y})$, $\overrightarrow{f(\overrightarrow{x})}$. Ou seja $\overrightarrow{f(\overrightarrow{x})}$ é dito não dominado se não existir nenhum $\overrightarrow{f(\overrightarrow{y})}$ que domine $\overrightarrow{f(\overrightarrow{x})}$

Pareto Ótimo: Dado um vetor solução $\overrightarrow{x} \in \Omega$, contendo funções objetivo $\overrightarrow{f(\overrightarrow{x})} = (f_1(\overrightarrow{x}), f_2(\overrightarrow{x})..., f_m(\overrightarrow{x})$ $\overrightarrow{x}$ é considerado ***Pareto Ótimo*** se somente se não existir $\overrightarrow{y} \in \Omega$, com o vetor representando as funções objetivo $\overrightarrow{f(\overrightarrow{y})} = (f_1(\overrightarrow{y}), f_2(\overrightarrow{y})..., f_m(\overrightarrow{y}))$, de tal forma que $\overrightarrow{f(\overrightarrow{y})} \preceq \overrightarrow{f(\overrightarrow{x})}$. Ou seja, diz que um vetor $\overrightarrow{x}$ é Pareto Ótimo se seu vetor objetivo é não dominado, que é quando as funções objetivo do vetor solução não pode ser potencializado simultaneamente.

Conjunto Pareto Ótimo: Para um problema de otimização multiobjetivo, é dado como Conjunto Pareto Ótimo P^*, um conjunto que possui as melhores soluções em Ω, ou seja, as soluções Pareto Ótimo definem o Conjunto Pareto Ótimo do problema. P^* é expressado por:

$$P^* := \{\overrightarrow{x} \in \Omega \mid \nexists \overrightarrow{y} \in \Omega \ , \ \overrightarrow{f(\overrightarrow{y})} \preceq \overrightarrow{f(\overrightarrow{x})}\} \tag{2.2}$$

Fronteira de Pareto: para cada solução em um conjunto Pareto Ótimo P^* possui uma imagem de um ponto não denominado no espaço de funções objetivo. Os vetores que não são dominados no espaço de funções objetivo formam a Fronteira de Pareto (PF, do inglês *Pareto Front*), que é definida por:

$$PF := \{\overrightarrow{f(\overrightarrow{x})} \mid \overrightarrow{x} \in P^*\} \tag{2.3}$$

2.2 Otimização com Muitos Objetivos

A literatura mostra um bom desempenho na solução de problemas com até três objetivos, no entanto existem problemas que envolvam quatro ou mais objetivos, denominados problemas de Otimização com Muitos Objetivos. Esse tem sido o foco de pesquisas nos últimos anos,

pois existem barreiras na sua otimização (DEB; JAIN, 2014a). Esses problemas enfrentam dificuldades diretamente relacionadas a sua dimensão, segundo Carvalho (2013), Saxena et al. (2013), os principais impasses enfrentados por esses problemas são:

Deterioração da habilidade de busca: Quando aumenta o número de funções objetivo, o número de soluções não dominada cresce também, ou seja, a habilidade dos algoritmos de busca é deteriorada, tornando-se praticamente uma busca aleatória ou guiada por aspectos de diversidade;

Alto custo computacional: Outra dificuldade está relacionada ao número de soluções para se aproximar a fronteira de Pareto. O número de soluções para se otimizar dado uma função objetivo cresce de maneira exponencial quando o número de funções cresce. Assim número de soluções necessárias para construir uma aproximação da fronteira de Pareto também cresce. Com isso, tornam-se mais complexo as estruturas e os mecanismos de diversidade utilizados pelos algoritmos de otimização;

Dificuldade de visualização gráfica: Quando a dimensão é superior a três, torna-se difícil a visualização dos pontos na fronteira de Pareto, já que técnicas de visualização simples só é possível uma visualização até no máximo tridimensional. Esse é um ponto negativo, pois a visualização é muito importante para a tomada de decisão.

A seguir serão apresentadas algumas possíveis soluções que estão sendo trabalhadas atualmente na literatura para diminuir as dificuldades que a computação enfrenta nos problemas de Otimização com Muitos Objetivos:

Redução de Objetivos: A redução da objetivos é uma alternativa para lidar com os problemas de muitos objetivos. A ideia geral é identificar os objetivos conflitantes que possam ser removidos sem que altere a qualidade dos dados no conjunto, ou seja, descartar os objetivos com quantidades desprezíveis de informações. Ao reduzir um conjunto de muitos objetivos para poucos objetivos (de muitos Objetivos para Multiobjetivos), as dificuldades enfrentadas pelos algoritmos diminuem, e algoritmos tradicionais conseguiriam resolver(SAXENA et al., 2013; BROCKHOFF, 2006). O uso dessa técnica é o principal foco desse trabalho, e será melhor explanada próxima seção.

Utilizar outras técnicas de avaliação de *fitness:* Um estudo que propõe a substituição da utilização da teoria de dominância de Pareto para avaliação de *fitness*, passando utilizar algoritmos que usam medidas de desempenho do conjunto de soluções, como hipervolume, epsilon e R2(ZITZLER; DEB; THIELE, 2000).

Uso de Estratégias de Decomposição: É uma abordagem que usa métodos decomposição de um problema multiobjetivo em sub-problemas. Assim tornando-se problemas escalares, que são capazes de serem otimizados por Algoritmos Evolucionários(ZHANG; LI, 2007).

Uso de Preferências: Visto que o número de soluções aumenta exponencialmente conforme cresce o número de objetivos, sendo assim uma boa estratégia é manter o foco em uma determinada região da fronteira de Pareto, usando indicadores de preferência (DEB; SUNDAR, 2006).

Controlar área de Dominância de Pareto: Nessa abordagem, utiliza métodos para controlar o grau de compressão e expansão da área da dominância de Pareto. Essa técnica trás uma aumento de performance no algoritmo evolucionário NSGA-II, no qual trás ótimos resultados na solução de problemas com Muitos Objetivos (SATO; AGUIRRE; TANAKA, 2007).

2.3 Redução de Objetivos

A maioria dos algoritmos evolutivos multiobjetivo disponíveis (MOEA - *Multi-Objective Evolutionary Optimization Algorithms*) para aproximar o conjunto de Pareto foi projetada e testada em problemas de baixa dimensionalidade, até três objetivos. Como foi discutido anteriormente, sabe-se que problemas com um número elevado de objetivos causam dificuldades adicionais em termos de qualidade da aproximação do conjunto de Pareto e do tempo de execução, e, o processo de tomada de decisão torna-se mais difícil quanto mais objetivos estão envolvidos. Brockhoff e Zitzler (2006) propôs trabalhar com um problema de otimização multiobjetivo, que busca encontrar um subconjunto menor de objetivos, preservando as informações, através de uma mudança na estrutura de dominância.

Brockhoff e Zitzler (2009) buscou resolver os problemas de deficiência na busca, alto custo computacional, dificuldade na tomada de decisão e visualização da fronteira de pareto. Primeiro foi investigado como fazer uma redução de objetivos baseados na remoção de objetivos conflitantes sem perda de informações, a segunda abordagem foi usar algoritmos heurísticos a fim de reduzir sistematicamente os objetivos, preservando a dominância do problema. Ou seja, reduzir a quantidade de objetivos ao máximo, mas preservando uma boa relação de dominância de Pareto. A partir da redução de objetivos, foi feita uma otimização, utilizando o algoritmo evolucionário NSGA-II. Através dos experimentos, foi possível chegar em resultados bem satisfatórios, assim concluindo que essa técnica pode ser um bom caminho a ser trabalhado em outras pesquisas.

Já Saxena et al. (2013) usou uma abordagem diferente. Foi desenvolvido um algoritmo que busca aplicar técnicas de aprendizado de máquina para identificar os objetivos essenciais e assim reduzir a dimensionalidade do problema, utilizando uma técnica baseada no PCA, que é um procedimento matemático que tem o pressuposto de que a estrutura de dados tridimensionais aglomerados pode ser revelada transformando de uma forma que o efeito do ruído e da redundância seja minimizado. Em outra palavras, o algoritmo busca identificar os objetivos essenciais e descartar os demais, assim transformando um problema complexo em um com dimensão menor e

mais fácil de ser resolvido com técnicas mais simples. Essa técnica será explorada nesse trabalho, sendo mais detalhada no capítulo 4.

2.4 Algoritmos Multiobjetivo

Nessa seção serão apresentados os principais conceitos dos algoritmos evolucionários NSGA-II (*Nondominated Sorting Genetic Algorithm II*) e NSGA-III (*Nondominated Sorting Genetic Algorithm III*). Esses serão explorados no trabalho para a otimização do problema.

2.4.1 Nondominated Sorting Genetic Algorithm II

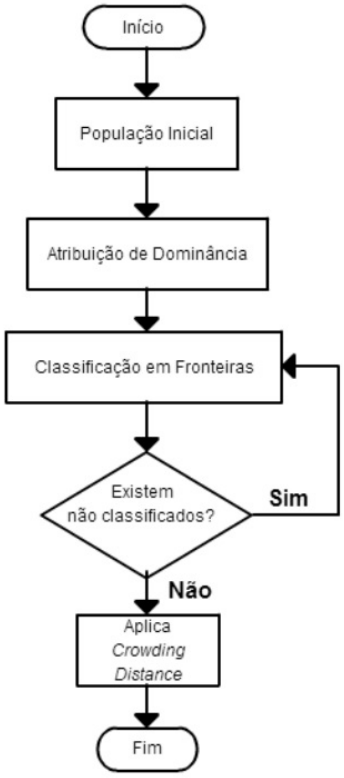
Na literatura, são encontrados trabalhos com bons resultados quando se utilizam algoritmos evolucionários (EAs) na otimização de problemas multiobjetivo, esses são denominados MOEAs (Multiobjetivo Evolucionary Algorithms)(HAJBABAIE; BENEKOHAL, 2015). Deb et al. (2000) utilizou-se de algoritmos evolucionários multi-objetivos (MOEAs) que geralmente possuem bons resultados aplicado nos Problemas de Otimização Multi-Objetivo (MOPs), foram testados cinco difíceis problemas, e o NSGA II em todos os supera os algoritmos PAES e SPEA, e assim concluiu que o NSGA-II é uma grande tendência em aplicações em um futuro próximo no âmbito de problemas Multiobjetivo.

O algoritmo NSGA-II é uma versão aprimorada do NSGA, que utiliza um procedimento rápido de classificação não dominada, uma abordagem de preservação elitista e um operador de nicho sem parâmetros. A principal diferença entre o NSGA e algoritmo genético simples é que realiza a operação estratificada, de acordo com as relações de dominação entre os indivíduos antes do operador de seleção. Os operadores de seleção, cruzamento e mutação são os mesmo de um GA simples. A estratégia de compartilhamento de *fitness* faz com que os indivíduos sejam distribuídos uniformemente, no NSGA também é mantida a diversidade da população. Em comparação com o NSGA, o NSGA-II é a extensão com os baixos requisitos computacionais, abordagem elitista e abordagem de compartilhamento sem parâmetros (SHEN; WANG; WANG, 2013).

O NSGA-II faz a classificação da população em níveis, denominados fronteiras. Essa classificação é feita da seguinte forma: a primeira, os indivíduos contidos nela representam as melhores soluções não-dominadas, na fronteira subsequente, os valores das soluções são menos significativo, a ultima fronteira estão as piores soluções. Dessa forma, podemos chegar a melhores resultados, com pontos mais próximos da fronteira de pareto e consequentemente melhor para os problemas Multiobjetivo.

A execução do NSGA-II é composta por dois componentes, a *Fast Non-Dominated Sorting* (classificação rápida por não dominância) e a *Crowding Distance* (distância de agrupamento). O primeiro é onde acontece a selação dos indivíduos por dominância.

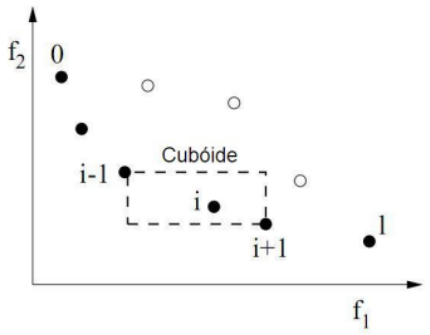
Figura 1 – Fluxograma de execução do NSGA-II



Fonte: (JORGE et al., 2014)

O segundo, é o *Crowding Distance*, que é o operador de diversidade do NSGA-II. Esse evita que haja concentração de soluções em um mesmo ponto ou região, também responsável por ordenar as soluções dentro da mesma fronteira. A ordenação é feita através da distância entre os indivíduos mais próximos, com o cálculo da distância média entre um ponto central i e de mais dois pontos de extremidades $(i+i)$ e $(i-1)$, a fim de espalhar os resultados ao longo da fronteira, formando um cubóide, representado pela figura 2.

Figura 2 – Fluxograma de execução do NSGA-II

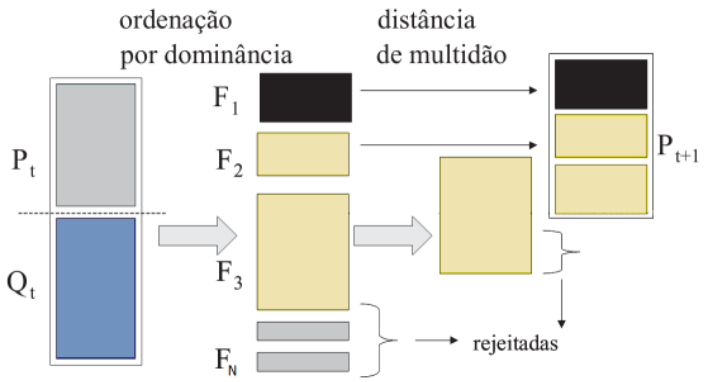


Fonte: (DEB et al., 2000)

O procedimento do NSGA-II, como mostra na figura 3 são classificadas duas populações,

uma população inicial de pais Pt, e posteriormente através de cruzamento, seleção e mutação a população de filhos Qt. A primeira fronteira $F1$ contém os indivíduos não-dominados e dominam as demais fronteiras $F2$,$F3$,...,Fn. No próximo passo, seleciona a partir da população que compõem a primeira fronteira, os indivíduos para completar a nova população. Caso uma fronteira não consiga ser totalmente inserida na população, o processo de *Crowding Distance* será responsável em selecionar quais indivíduo irão compor a nova população.

Figura 3 – Procedimento padrão do NSGA-II



Fonte: Adaptado de (DEB et al., 2000)

2.4.2 Nondominated Sorting Genetic Algorithm III

O NSGA-III (*Nondominated Sorting Genetic Algorithm III*), proposto por Deb e Jain (2014b), é uma nova abordagem baseada nos mesmos mecanismos de seleção do NSGA-II (*Nondominated Sorting Genetic Algorithm II*). A ideia é usar pontos de referência previamente definidos ou gerados através do sistema, e não tirando a característica de usar a Dominância Pareto. Esse demonstra uma boa eficácia na resolução de problemas de otimização de dois objetivos a 15 objetivos.

Os pontos de referências é um conjunto de pontos no espaço de objetivos que leva o processo de busca para a melhora na diversidade e convergência do conjunto de soluções. A estrutura proposta pelo algoritmo permanece a mesma do NSGA-II, só que o NSGA-III não utiliza o *Crowding Distance* na hora da seleção, mas o fornecimento desses pontos de referências citados anteriormente.

Como o NSGA-III utiliza os pontos de referências previamente fornecidos ao algoritmo, acaba priorizando as soluções não-dominadas e também trás os indivíduos associados a cada ponto de referência individualmente. Assim, faz com que os resultados do algoritmo tenha tendência à encontrar soluções próximas a fronteira de pareto relacionadas aos pontos fornecidos.

No trabalho de Deb e Jain (2014a) foi desenvolvido o NSGA-III, que usa uma abordagem baseada em pontos de referência, para tentar resolver problemas de otimização com muitos

objetivos. O algoritmo foi aplicado a problemas entre três e quinze objetivos, e em todos os testem o algoritmo NSGA-III obteve sucesso repetidamente em várias execuções. Em problemas com dimensões maiores, o algoritmos enfrentam uma dificuldade cada vez maior em manter a diversidade e convergir para a frente ótima de pareto, mas essa abordagem tem um diferencial, que é a ajuda na preservação da diversidade, usando um conjunto de pontos de referência bem distribuído, tornando possível encontrar uma solução ótima de pareto. Ou seja (DEB; JAIN, 2014a) abordou no artigo as e tentou resolver as dificuldades relacionadas à solução de problemas de otimização de muitos objetivos e sugeriu um algoritmo evolucionário que fosse viável para problemas de muitos objetivos e conseguiu demonstrar ótimos resultados. Assim podemos claramente afirmar que o uso e aplicação do NSGA-III resultados concretos e pode ser aplicados em outros problemas desafiadores no mundo atual.

2.5 Trabalhos Relacionados

(STEVANOVIC et al., 2015) trabalhou com um problema tridimensional, onde ele tentou abordar a otimização nos campos de mobilidade, segurança e meio ambiente. Este vem inovando nessa área, pois oferece uma solução de três objetivos, algo que na literatura é comum trabalhar com até dois objetivos. Utilizou o algoritmo evolucionário NSGA-II na otimização, para identificar as melhores soluções. Foi utilizado um cenário contendo 5 cruzamentos com semáforos, e com a contribuição de um simulador de trânsito se fez possível a virtualização de um cenário baseado na cidade de West Valley City. Como resultado, houve sucesso na busca de uma sincronização ótima, proporcionando um equilíbrio entre mobilidade, segurança e meio ambiente. O trabalho teve como limitação a quantidade de objetivos trabalhados, apesar de trabalhar com um problema de três dimensões, ainda não é o ideal para representar um cenário próximo do real.

(KHAMIS; GOMAA, 2014) desenvolveu um sistema adaptativo de aprendizagem multi-objetivo para o controle de sinal de trânsito, usou como embasamento uma estrutura multi-agente cooperativa à base de veículos. Utilizou informações de aceleração e desaceleração em situações como os estados semafóricos, verde e vermelho, acarreta no processo de tomada de decisões da nova configuração semafórica adequada, utilizando interpretação de probabilidade Bayesiana para estimar os parâmetros. Obteve bons resultados, conseguiu mostrar que o controlador de sinal de tráfego proposto supera outros controladores usando o padrão de tráfego do centro da cidade com demandas concorrentes. Sua limitação foi não tratar de acontecimentos especiais como acidentes ou interdição de vias, onde uma desaceleração poderá ser entendida como um estado semafórico vermelho, tomando assim uma decisão indevida.

Em (SUN; BENEKOHAL; WALLER, 2003) utilizou um Algoritmo Genético na solução do problema de otimização de temporização de sinais de intersecção multi-objetivo. Três problemas foram definidos, e com uso do NSGA II tentar minimizar os objetivos: atraso mé-

dio e o número médio de paradas, usando o tempo verde efetivo em cada fase do ciclo como variável de projeto. O primeiro problema possui um padrão de tráfego uniforme sem restrição de comprimento de ciclo, outro também com trafego uniforme, mas com uma restrição de comprimento mínimo de ciclo, e a terceira com padrão de tráfego estocástico com a restrição de comprimento de ciclo. O cenário foi projetado para uma interseção isolada acesso permitido para a esquerda. As aproximações matemáticas dos parâmetros de Pareto resultantes são apresentadas para avaliar a distância entre vários objetivos e assim fornecer as melhores temporizações para todas as situações da interseção no semáforo. Portanto, a partir dos resultados que o algoritmo genético multi-objetivo tem uso potencial na otimização do sincronismo do sinal de interseção. A limitação desse trabalho foi de abranger apenas dois objetivos.

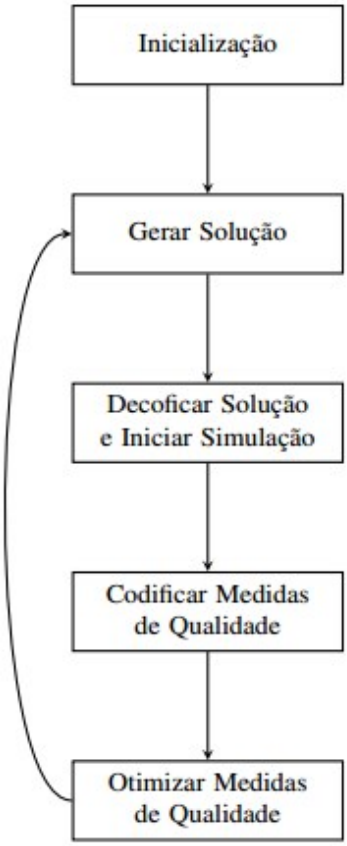
No trabalho de (KWASNICKA; STANEK, 2006) é proposto um método de otimização de semáforos com base em um algoritmo genético e o uso de uma simulação microscópica. Seu objetivo é minimizar o tempo de viagem e maximizar quantidade de veículos que circulam no local. O trabalho utiliza duas configurações para realizar os experimentos, uma com e outra sem atuadores dentro da malha viária. Ao utilizar atuadores encontram resultados mais precisos, sem um aumento significativo do custo computacional. Em ambas as configurações testadas os resultados foram alcançados. As filas foram esvaziadas rapidamente e não existiram situações onde os veículos não foram capazes de entrar na área que foi estudada. Apesar de bons resultados, a limitação encontrada foi que a proposta do sistema permite otimizar apenar um único objetivo.

2.5.1 Sincronização de Semáforos como um Problema de Otimização com Muitos Objetivos

(MATOS, 2017) modelou o problema da sincronização de semáforos como um problema de otimização com muitos objetivos. Diferentemente dos outros trabalhos, utilizou seis funções objetivos: tempo de viagem e paragem, número de veículos que chegam ao seu destino, velocidade média global e fluxo de tráfego, emissão de dióxido de carbono. Trabalhar com muitas funções objetivos, faz com que o trabalho desenvolvido se assemelhe ao máximo a situações reais. O problema sincronização de semáforos é modelado como um problema de otimização, com a utilização de um simulador de tráfego, com esse se faz possível construir uma representação computacional, que através de uma configuração semafórica se obtenha as medidas de qualidade. O sistemas desenvolvido utilizou o SUMO como simulador. Para otimizar as medidas, os algoritmos usados foram o NSGA-II e NSGA-III, que são algoritmos evolucionários que tem ótimos resultados para resolver problemas multiobjetivo. Para garantir a comunicação dos algoritmos de otimização com o simulador, foi utilizado um framework chamado jMetal, que tem nele contido os algoritmos de otimização.

A organização do sistema desenvolvido por Matos (2017) está representado como mostra a Figura 4. E os seguintes passos foram seguidos:

Figura 4 – Representação esquemática da otimização semafórica



Fonte: MATOS, Saulo (2017)

- Inicialmente, parâmetros de configurações foram inseridos no simulador (SUMO foi o escolhido), como: mapas, semáforos, demandas de veículos, rotas a serem seguidas pelos veículos e quais medidas de qualidade serão utilizadas. Essas configurações são feitas através da edição arquivos no formato XML, como podemos visualizar nas imagens 10, 5 e 6. Com tudo configurado, inicia as simulações no SUMO para se extrair as medidas de qualidades.

Figura 5 – Configuração de fases de um semáforo

```
  <tlLogic id="10" type="static" programID="0" offset="0">
  <phase duration="53" state="GGrrGGrr" />
  <phase duration="4" state="yyrryyrr" />
  <phase duration="45" state="rrGGrrGG" />
  <phase duration="4" state="rryyrryy" />
</tlLogic>
```

Fonte: O autor

Figura 6 – Configuração de fluxo de veículos

```
<flows>
  <flow id="E14_26" from="E14" to="E26" begin="0" end="3600" number="900"
  ↪  type="CarA" />
  <flow id="E27_15" from="E27" to="E15" begin="0" end="3600" number="2700"
  ↪  type="CarA" />
  <flow id="E28_40" from="E28" to="E40" begin="0" end="3600" number="1500"
  ↪  type="CarA" />
  <flow id="E41_29" from="E41" to="E29" begin="0" end="3600" number="1100"
  ↪  type="CarA" />
</flows>
```

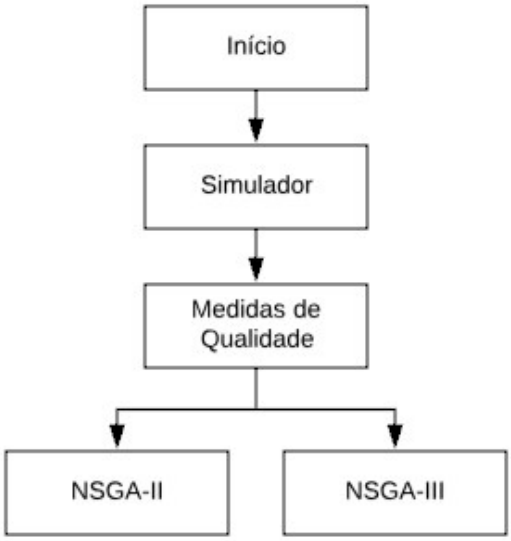
Fonte: O autor

- O próximo passo, é constituído pela modelagem do problema de sincronização de semáforos como problema de otimização. Para tal fim, é representando computacionalmente uma possível combinação dos tempos semafóricos através de um vetor, que vai representar os tempos de fases a serem otimizadas. A partir desse ponto, os algoritmos de otimização serão utilizados para manipular esses vetores com a finalidade de encontrar temporizações ideais que otimizem as medidas de qualidades. Como base de dados inicial, foram utilizados seis configurações de *benchmark*.

- Essa etapa consistem em otimizar as medidas de qualidades, para isso, como os algoritmos evolucionários NSGA-II e NSGA-III são encontrados na literatura com ótimos resultados nesses problemas, esses serão utilizados na otimização. Para tal fim, será utilizado um *framework*, denominado *jMetal*, que é composto por algoritmos metaheurísticos em Java orientado a objetos, usado para otimização multiobjetivo. O processo de otimização é feito em conjunto com o simulador, onde ao ser gerado uma combinação semafórica pelos algoritmos, uma simulação é feita, e as novas medidas de qualidades são geradas. Esse processo é realizado inúmeras vezes até que encontre combinações que mais se aproximem a fronteira de pateto. A figura 7 ilustra todo o processo do sistema de otimização.

- Para melhorar ainda os resultados, um ultimo passo foi desenvolvido. Foi aplicado um algoritmo de redução de dimensionalidade chamado DRP (do inglês *Dominance relation preservation*), baseado em preservar as relações de dominância das soluções não dominadas, conseguiu reduzir a quantidade de objetivos. O mesmo procedimento de otimização foi feito com as medidas reduzidas, onde atingiu ótimas soluções. A limitação encontrada na pesquisa foi que ao propor uma sincronização que se aproxime de um ambiente real, trabalhar com apenas seis medidas se torna insuficiente, pois num ambiente verdadeiro existem qualidades que são influenciadas pelo trânsito.

Figura 7 – Fluxograma do sistema de otimização



Fonte: O autor

No fim de todo o processo feito por Matos (2017), uma base de dados é gerada com as melhores soluções possíveis para as configurações utilizadas (cenário, mapa, demanda de veículos e etc). Essa base de dados é utilizada como entrada no sistema desenvolvido nesse trabalho, e todos os detalhes serão discutidos no próximo capítulo.

É possível identificar algumas limitações no trabalho proposto por Matos (2017). Embora tenha obtido bons resultados, existem alguns pontos que podem ser melhorados. O primeiro é que apesar de existir na literatura outras técnicas de redução de objetivos, só foi explorada no trabalho somente uma. Outra melhoria seria a utilização de mais funções objetivos, já que o simulador oferece muitas funções e no trabalho só foram exploradas 6. Já no sistema proposto nesse trabalho, que será apresentado no próximo capítulo, é feito a aplicação de mais técnicas de redução de objetivos e serão utilizadas 12 funções objetivos.

3

Sincronização de Semáforos

A organização desse capítulo terá a seguinte estrutura: seção 3.1 é apresentado os conceitos de sistemas de Sistemas de Transportes Inteligentes (ITS - do inglês *Intelligent Transportation System*) e Gestão de Tráfego; na subseção 3.1.1 é dissertado sobre Sincronização de Semáforos; como subseção 3.1.2 é abordado Sincronização de Semáforos como problemas de Otimização com Muitos Objetivos; seção 3.2 é apresentado conceitos de simuladores de trânsito; seção 3.3 são apresentados os trabalhos relacionados existentes na literatura.

3.1 Sistemas de Transportes Inteligentes e Gestão de Tráfego

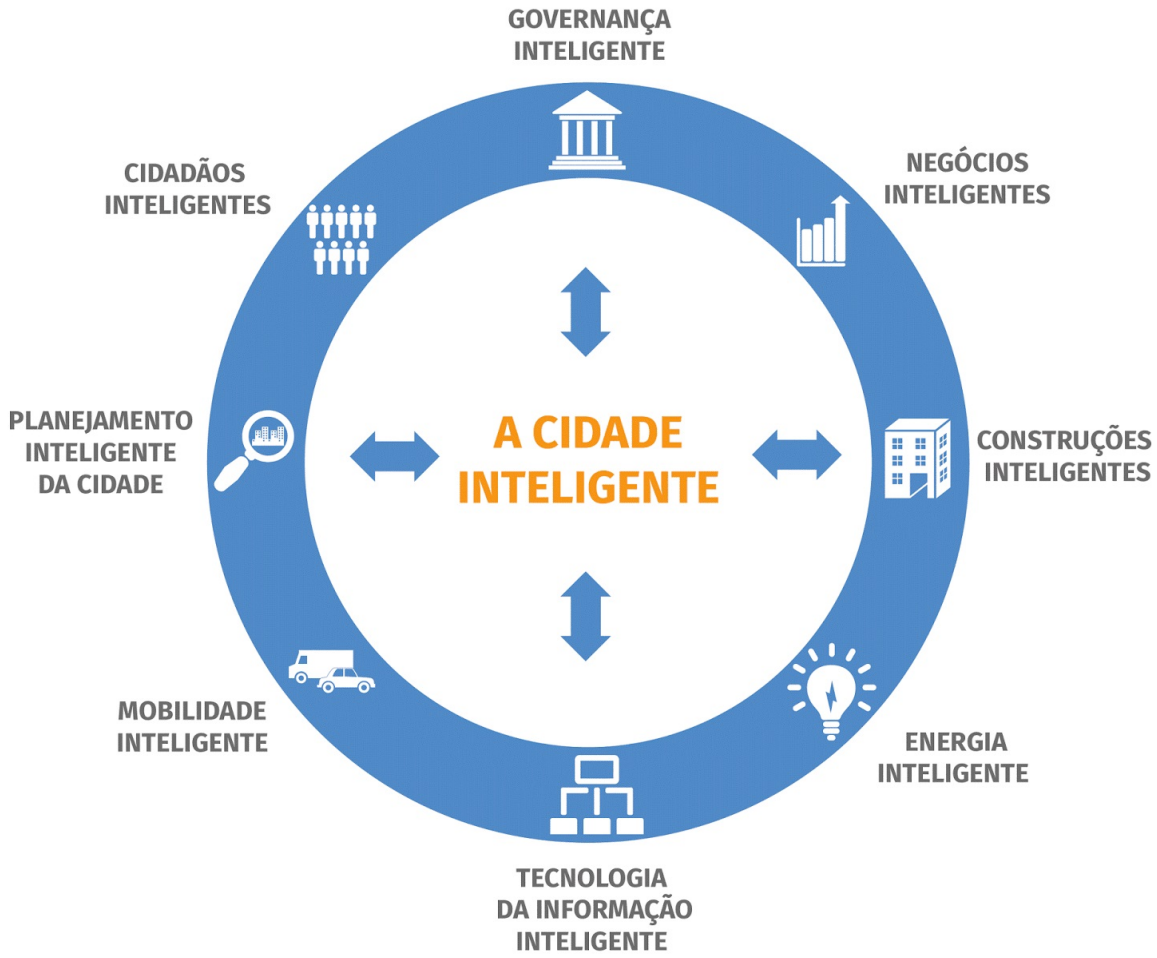
Com o crescimento dos centros urbanos das cidades modernas e o aumento da quantidade de veículos, os problemas de mobilidade urbana vêm se agravando cada vez mais. No quadro atual, existem faixas exclusivas, mudanças nos sentidos das vias, viadutos, passarelas, túneis, novos semáforos e outras soluções. Atualmente, os engenheiros de trânsito ajustam a temporização dos semáforos, seguindo um conjunto de padrões de tráfego, verificados em um passado recente, n tentativa de definir um tempo semafórico ideal para aquele cruzamento, mas ainda não alcançam a solução total do problema (MCKENNEY; WHITE, 2013).

Uma solução para esses desafios é seguir o princípio de que as cidades sejam mais eficientes, utilizando recursos tecnológicos ao seu favor. Uma abordagem inovadora é a de Cidades Inteligentes, que tem a premissa de funcionar de uma forma sustentável e inteligente, integrando todas as suas infraestruturas e serviços num sistema homogêneo, contando com dispositivos de monitoramento e controle que age de forma inteligente para garantir a sustentabilidade e eficácia (HANCKE; JR et al., 2012).

Atualmente a cidades do mundo tendem a implantar sistemas inteligentes para gestão de vários seguimentos diferentes, atendendo várias necessidades e melhorando a forma de itera-

ção com os cidadãos. Os sistemas inteligentes tendem a em otimizar componentes em vários setores da economia, tais como: saúde, negócios, construções, transporte, energia, governança, agricultura, educação, indústria, tecnologia da informação e comunicação (TIC), cultura, militar e justiça, entre outros como é ilustrado na Figura 8; mas também tendem a serem mais eficientes aos processos urbanos. Visto isso, os governos e empresas privadas tendem a investir nesse seguimento, criando assim um cenário competitivo e de fundamental importância para o desenvolvimento e economia(HERRERA-QUINTERO et al., 2015).

Figura 8 – Cidades inteligentes.



Fonte: SOUZA (2016)

Dentro desse contexto, temos a implantação do monitoramento de tráfego, que usam sensores e atuadores para coletar informações do trânsito. Esses dados são enviados para um sistema de informações central, onde decisões inteligentes baseadas nesses dados podem ser tomadas. Os Sistemas de Transporte Inteligente (ITS) são muito importantes nos conceitos de cidades inteligentes, contribuindo de forma expressiva nessa área(HERRERA-QUINTERO et al., 2015).

Com a transição de sistema de transporte tradicional para um sistema que usa da tecnologia da informação, é possível mudar completamente as condições de tráfego. Visto que, pode alcançar melhorias aos usuários como melhor fluidez no seu percurso, redução de tempo de viagem, consumo de combustível e emissão de poluentes. ITS é efetivamente um trabalho em conjunto feito com a tecnologia da informação, a tecnologia de contato eletrônico, a tecnologia de comunicação de dados e a tecnologia de processamento computadorizado, aplicadas no campo da transporte para construir um sistema de gestão composta por pessoas, estradas e veículos (PENG; JIANG-PING; JING, 2009; SEREDYNSKI; MAZURCZYK; KHADRAOUI, 2013).

3.1.1 Sincronização de semáforos

A sincronização de semáforos vem sendo utilizada atualmente, com o intuito de tentar melhorar alguns impasses decorrentes desse cenário de crescimento gradativo. A sincronização estima aumentar a fluidez do trânsito e a capacidade de interseção em cruzamentos, diminuir as paradas e atrasos, bem como, contribuir para a redução do tempo de viagem, consumo de combustível e emissões de gases poluentes e entre outros benefícios(ABDOOS; MOZAYANI; BAZZAN, 2011).

Segundo Oliveira (2005) uma sincronização é alcançada quando pelo menos dois semáforos executados em conjunto façam com que um veículo realize o percurso completo sem paradas, mas nem sempre isso será possível. Vários fatores influenciam diretamente nesse resultado, entre eles podemos encontrar um alto número de veículos em diversos sentidos e situações imprevisíveis como alagamentos e acidentes.

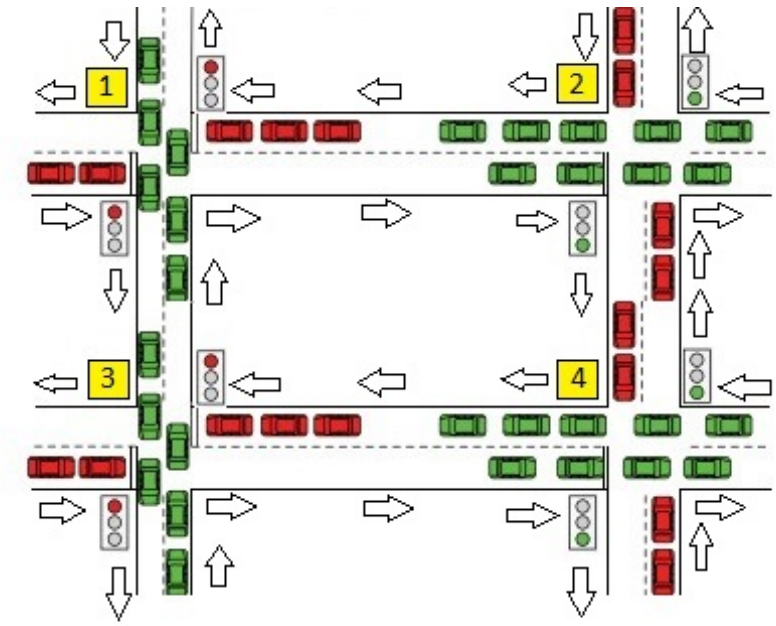
A maioria das cidades atualmente tendem a utilizar um tipo de configuração semafórica, utilizam uma configuração fixa, que são calculados através de padrões de fluxo de tráfego que se repetem em algumas situações rotineiras (feriados, horário de pico), pois é mais barato financeiramente. Já os modelos automáticos de controle de sinal, são projetados para alterar as configurações semafóricas em tempo real, através da a situação do tráfego. Mas o tráfego veicular tem uma característica dinâmica, que podem ocorrer situações de alteração de tráfego individuais que não são previstas. É crescente o número de trabalhos encontrados nesse seguimento, mas vários problemas como o alto custo operacional são enfrentados, fazendo com que seja mais difíceis de serem implantados(GARCIA-NIETO; OLIVERA; ALBA, 2013).

No contexto de sincronização de semáforos, existem alguns conceitos que são essenciais para o entendimento, são ele: semáforo, componente eletrônico que fornece informações aos condutores de veículos e aos pedestres através de indicações luminosas; onda verde, é quando a sincronização de semáforos proporciona que os veículos passem por maior quantidade de semáforos no sinal verde (estado “aberto”) ao longo de uma via; estágio, é cada intervalo de tempo em que o sinal verde ou vermelho não se altera; fase, é o conjunto de sequencias de indicações luminosas; ciclo, é definido quando todos os grupos semafóricos tenha passado pelo sinal verde pelo menos uma vez; defasagem, é o tempo necessário de espera entre um semáforo

e o outro subsequente para que seja possível acontecer a onda verde; rota, é o percurso em que o veículo vai percorrer no cenário (MATOS, 2017).

A Figura 9, é um exemplo de um cenário com quatro cruzamentos, enumerados de 1 até 4, nela podemos notar que existem várias direções de escolha para cada veículo, indicados pelas setas e várias fases semafóricas, exibidos nas cores verde representando o siga e vermelho o pare.

Figura 9 – Rede semafórica com quatro cruzamentos.



Fonte: Adaptado de LI, Yan et al.(2013)

As pesquisas atuais buscam encontrar um plano semafórico ideal, mas isso não é uma tarefa fácil. Segundo (HAJBABAIE; BENEKOHAL, 2015) a otimização em sincronização se semáforos de classifica como problema NP-Completo.

3.1.2 Sincronização de Semáforos como problemas de Otimização com Muitos Objetivos

A otimização na sincronização de semáforos consiste em otimizar o ciclo de luzes de trânsito, a sequência de estados verde, amarelo e vermelho, que um sinal de trânsito utiliza no seu sistema. Para realizar essa otimização, são analisadas diferentes interseções rodoviárias e semáforos, com vários fluxos. Melhores temporizações serão procuradas para que medidas de qualidade de tráfego possam ser melhoradas, dentre essas medidas de qualidades podemos citar: consumo de combustível, emissão de poluentes, tempo de espera, quantidade de paradas, velocidade média, tempo de viagem, emissão de ruídos por veículo, total de veículos. Levando

em consideração, que quanto mais medidas de qualidades forem trabalhadas, mais próximo de um cenário real será o estudo.

É possível modelar o problema da sincronização de semáforos como um problema de otimização, representando virtualmente possíveis combinações de semáforos em um vetor. Com o uso de um simulador de tráfego, através do vetor de combinações de tempos, são obtidas as medidas de qualidade oriundas do próprio simulador. Cada umas dessas medidas que serão otimizadas, podemos modelar como funções objetivo num problema de otimização. Como o a quantidade de objetivos extraídas do simulador é superior a três, considera-se o problema como de otimização com Muitos Objetivos (KHAMIS; GOMAA, 2014).

3.2 Simulador

Os simuladores de trafego são essenciais na modelagem do problema, com o seu uso é possível prever comportamentos de tráfego em diversas situações do dia-a-dia. Em questão do funcionamento, os simuladores são trabalhados da seguinte forma: previamente são configuradas as rotas, a quantidade de veículos, os cenários, e as medidas de qualidade que queremos como saída. Posteriormente são feitas as simulações, e, a partir de então, se obtêm os dados de saídas (medidas de qualidades), que são diretamente relacionados ao comportamento daquele cenário nas situações diversas. Simulações de tráfego facilitam a avaliação de mudanças de infraestrutura na estrada antes mesmo de serem executadas. Por exemplo, uma otimização de sincronização de semáforo pode ser testada em uma simulação antes de ser implantada no mundo real (GRIGGS et al., 2015).

Na literatura existem muitos trabalhos que utilizam o simulador SUMO (do inglês *Simulation of Urban Mobility*). Foi desenvolvido por funcionários do Instituto de Sistemas de Transporte do Centro Aeroespacial da Alemanha e ficou disponível desde 2001. O SUMO permite a modelagem de sistemas de tráfego incluindo rotas, cruzamentos, semáforos, veículos rodoviários, transporte público e pedestres. Além disso está disponível várias ferramentas que lidam com rotas, redes viárias, cálculos de tempos e distâncias, consumos e emissões (baseados em rotas e viagens). (KRAJZEWICZ; RÖSSEL, 2006).

Todas as configurações são feita através de arquivos no formato .XML, configurações de Rotas, quantidade de veículos, e características individuais de cada veículos como: tipo, tamanho, velocidade máxima, aceleração, desaceleração, cor, como mostra no código 10. Informações sobre os ciclos de semáforos e cruzamentos, também são configuradas através de arquivos. Como saída, também salvo em arquivos, extraímos as medidas de qualidades referentes as viagens ocorridas nas simulações, dentre elas estão: horário do inicio da viagem de cada veículo, número de paradas, tempo de espera por veículo, duração da viagem, consumo de combustível e emissão de gases poluentes, como podemos ver no código 11.

Segundo a documentação do SUMO (KRAJZEWICZ; RÖSSEL, 2006), segue as princi-

Figura 10 – Arquivo de configuração dos veículos

```
<vType id="CarA" length="5.00" minGap="2.50" maxSpeed="60.00" color="red">
    <carFollowing-Krauss accel="10.00" decel="4.00" sigma="0.50"/>
</vType>
<vehicle id="E0_12.0" type="CarA" depart="0.00">
    <route edges="E0 E4 E8 E12"/>
</vehicle>
<vehicle id="E11_47.0" type="CarA" depart="0.00">
    <route edges="E11 E25 E39 E47"/>
</vehicle>
<vehicle id="E13_1.0" type="CarA" depart="0.00">
    <route edges="E13 E9 E5 E1"/>
</vehicle>
```

Fonte: O autor

Figura 11 – Arquivo de saída das simulações

```
<tripinfo id="E44_6.0" depart="0.00" departLane="E44_0" departPos="5.10"
 ↪  departSpeed="0.00" departDelay="0.00" arrival="17.00" arrivalLane="E6_0"
 ↪  arrivalPos="191.95" arrivalSpeed="57.03" duration="17.00" routeLength="794.90"
 ↪  waitSteps="0" timeLoss="3.53" rerouteNo="0" devices="tripinfo_E44_6.0
 ↪  emissions_E44_6.0" vType="CarA" speedFactor="1.00" vaporized="">
     <emissions CO_abs="18445.942455" CO2_abs="745420.666308" HC_abs="102.995308"
      ↪  PMx_abs="20.779633" NOx_abs="339.547010" fuel_abs="320.420306"
      ↪  electricity_abs="0"/>
  </tripinfo>
```

Fonte: O autor

pais funcionalidades:

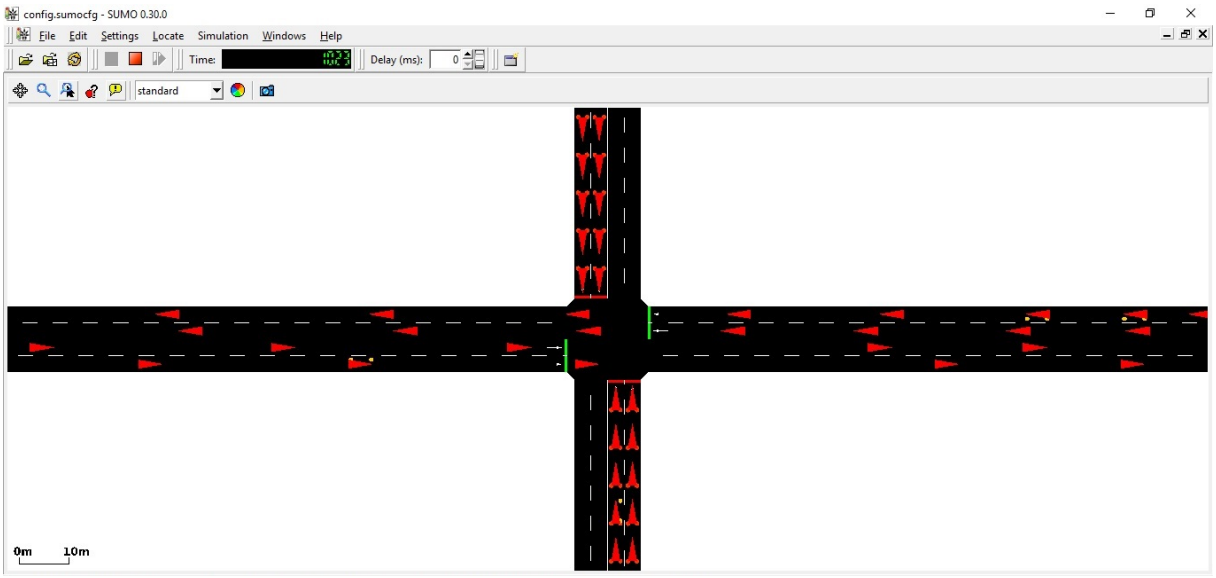
- Fluxo de trabalho completo (importação de rede e rotas, simulação);
- Simulação microscópica completa, próximo de um cenário real;
- Movimento de veículo livre de colisão;
- Diferentes tipos de veículos (passeio, transporte de carga e passageiros);
- Ruas com várias faixas com mudança de faixa;
- Regras de direito de passagem com base em junção;
- Hierarquia de tipos de junção;

- Uma interface gráfica para o usuário, openGL e rápida;
- Gerencia redes com várias ruas e contornos;
- Velocidade de execução rápida (até 100.000 atualizações de veículo por segundo considerando uma máquina de 1 GHz);
- Interoperabilidade (via arquivos .XML) com outro aplicativo no momento da execução;
- Rotas microscópicas, onde cada veículo tem a sua própria;
- Uso de heurística para definir valores não configurados;
- Boa compatibilidade com as principais distribuições do Linux e Windows;
- Bibliotecas em C++ são utilizadas.

Segue também os módulos disponíveis:

- SUMO - Simulação microscópica sem visualização, aplicação de linha de comando;
- GUISIM - Simulação microscópica com uma interface gráfica do usuário;
- NETCONVERT - Importador e gerador de rede, lê redes viárias de estradas de diferentes formatos e as converte no formato SUMO;
- NETGEN - Gera redes abstratas para a simulação SUMO;
- DUAROUTER - Calcula as rotas mais rápidas através da rede, importando diferentes tipos de descrição de demanda;
- JTRROUTER - Calcula rotas usando porcentagens de giro em interseções;
- DFROUTER - Calcula rotas com uso de dados do detector;
- OD2TRIPS - Decompõe matrizes de origem e destino em viagens de veículo único;
- POLYCONVERT - Importa pontos de interesse e polígonos de diferentes formatos e os traduz em uma descrição que pode ser visualizada pelo GUISIM (simulação com interface gráfica);
- MAROUTER - atribuição macroscópica de usuários baseada nas funções de capacidade;
- NETEDIT - Editor visual para redes, semáforos, detectores e outros elementos;
- AdditionalTools - Várias soluções para problemas diferentes podem ser cobertas por essas ferramentas.

Figura 12 – SUMO GUI com uma simulação em andamento



4

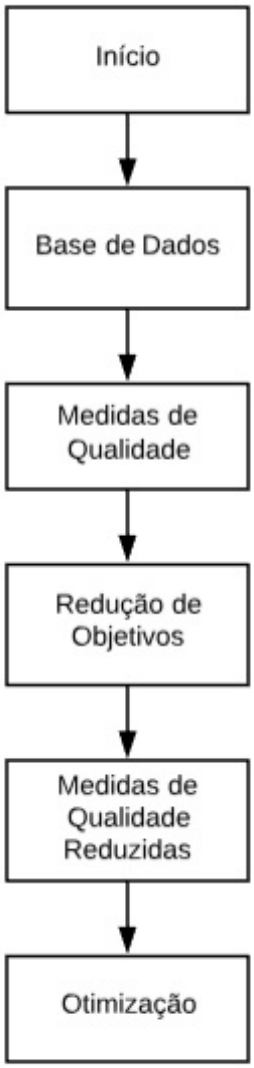
Técnicas de Redução de Dimensionalidade no Problema de Sincronização de Semáforos

A ideia geral desse trabalho, é através da base de dados construída pelos algoritmo de otimização anteriormente, utilizar técnicas de aprendizagem de máquina de redução de objetivos, para identificar os objetivos que possuem a maior quantidade de informações do problema, e desprezar os demais, assim diminuindo a complexidade do problema. Após o processo de redução de objetivos, o processo de otimização será executado, utilizando como entrada o conjunto menor de objetivos, afim de atingir uma melhor performance nos algoritmos evolucionários, e tentar resolver o problema de sincronização de semáforos. Dois processos de redução de objetivos são utilizados, um com técnica linear e o outro com técnica não linear, dessa forma, serão apresentados na próxima seção.

4.1 Sistema Desenvolvido de Redução de Objetivos

Inicialmente foi implementado um algoritmo baseado no trabalho de (SAXENA et al., 2013), que usa uma técnica de aprendizagem de máquina para identificar os objetivos conflitantes e fazer a redução da dimensão do problema (diminuir a quantidade de funções objetivo). O algoritmo se baseia nos conceitos de análise de componentes principais (PCA), que é um procedimento matemático que tem o pressuposto de que a estrutura de dados tridimensionais aglomerados pode ser revelada transformando de uma forma que o efeito do ruído e da redundância seja minimizado. Para isso acontecer, um dado X é projetado de modo que sua estrutura de

Figura 13 – Fluxograma do sistema desenvolvido



Fonte: O autor

correlação seja mais fielmente preservada, na medida em que PCA projeta X nos autovetores da matriz de correlação de X. Tendo como resultado conjunto de valores de variáveis linearmente não correlacionadas chamadas de componentes principais(SAXENA et al., 2013).

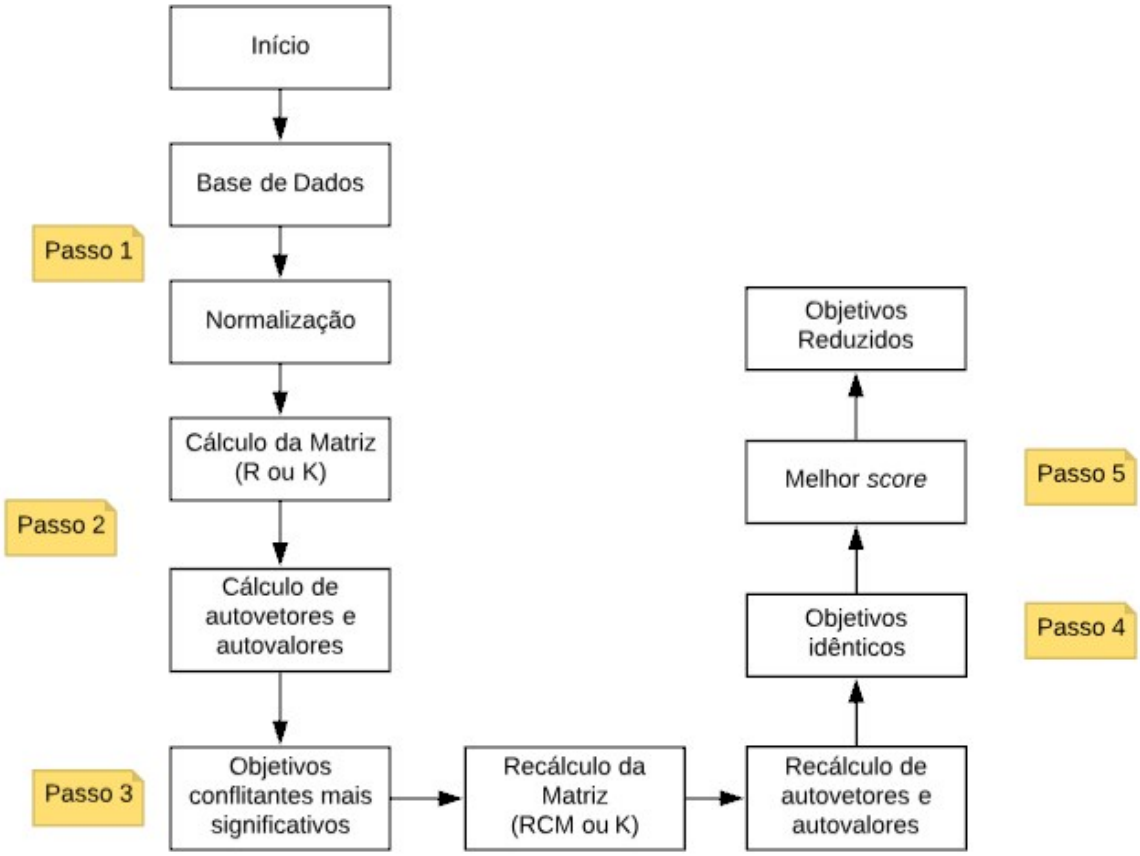
No algoritmo de redução desenvolvido, é projetada uma matriz através do dados iniciais, e a partir dela serão feitos cálculos e analises para identificar os objetivos que tem a maioria das informações do problema, e descartar as medidas que tem o mínimo de informações próximos de ser desprezíveis. Essa matriz é gerada de duas formas distintas, usando método linear(K-PCA) e não-linear(NL-PCA). A obtenção de forma linear, é feita com o calculo da matriz de correlação, como é feito no PCA, e o método não-linear, é obtida através do cálculo de uma matriz de kernel, que em (SAXENA et al., 2013), o método utilizado foi o MVU (do inglês *maximum variance*

unfolding), que se adaptar ao tipo de problema trabalhado.

Já no método desenvolvido nesse trabalho, são utilizados vários kernels distintos, como o polinomial, o gaussiano e o hiperbólico. Todo o sistema foi implementado na linguagem de programação Java, integrado com uma biblioteca de mineração de dados denominada WEKA. Todo o processo do sistema desenvolvido pode ser visualizado no fluxograma 13.

Com a base de dados como entrada, o algoritmo realiza os seguintes passos de análise dos dados para identificar os objetivos essenciais, esses podem ser acompanhar através do fluxograma 14 e do pseudocódigo 15:

Figura 14 – Fluxograma do sistema de Redução



Fonte: O autor

Passo 1: O primeiro passo consiste na preparação dos dados a serem trabalhados. Os dados iniciais que são usados, são as funções objetivos são extraídas do simulador através de arquivos, que para a utilização das mesmas pelo WEKA foi realizada uma conversão para o formato de extensão .arff, formato aceito pela biblioteca, como é mostrado na linha 4 do pseudocódigo 15.

Para evitar problemas derivados do uso de funções objetivos com diferentes unidades de medidas e escalas, foi feita uma normalização dos dados. O processo de normalização

Figura 15 – Pseudo-código do Algoritmo L-PCA e NL-PCA

```
1  Entrada:
2      Medidas de qualidades oriundas das execuções dos algoritmos NSGA-II e NSGAII;
3  Inicio:
4  Ler os dados transformando-os em uma matriz;
5  Normaliza os dados (v - ValorMínimo) / (ValorMáximo - ValorMínimo);
6  R (ou K) = Cálculo da matriz de correlação(ou matriz Kernel) através da matriz de
    ↪  dados;
7  Q = Cálculo dos autovetores de R (ou K);
8  L = Cálculo dos autovalores de R (ou K);
9  Threshold = 0.997;
10 Truncar Q  relacionado ao somatório de L > Threshold
11 Indices[ ];
12 for linhas de Q;
13    for colunas de Q;
14       Seleciona maior valor absoluto;
15       Indices[ ] = Indice do valor absoluto;
16       Identifica o sinal do maior valor;
17       if existir valores com sinais opostos ao maior valor then
18          Indices[ ] = Indices desses valores;
19       else
20          Seleciona o segundo maior valor absoluto;
21          Indices[ ] = Indice desse valor;
22    Remove da matriz de dados as colunas equivalentes aos não índices coletados;
23    RCM = Cálculo da matriz de correlação(ou matriz Kernel) dos dados da matriz
       ↪  Reduzida;
24     Threshold2 = 0.954;
25     Truncar Q  relacionado ao somatório de L > Threshold;
26     Calcula Tcor;
27    for linhas de RCM;
28       for colunas de RCM;
29            Seleciona os itens que são > Tcor;
30                    identicos[ ]    =  itens;
31                    Calcula Score dos itens;
32                    Seleciona o Score o item que tem maior Score;
33                    Remove da matriz de dados as colunas equivalentes aos não índices
                       ↪  coletados;
34 fim
```

Fonte: O autor

consiste em transformar os dados e deixa-los em uma escala entre 0 e 1. Procedimento equivalente à linha 5 do pseudocódigo 15.

Passo 2: Essa etapa representa o cálculo das matrizes, onde a partir dos dados iniciais e do conjunto original de funções objetivos F normalizados, transformamos esse dados em uma matriz. O cálculo foi feito utilizando a biblioteca WEKA integrada com o Java. As matrizes

que foram calculadas foram, a matriz de correlação R(método linear) ou a matriz de kernel K (método não-linear). A partir dessas, foram calculados os autovetores e autovalores. Cada posição de cada autovetor corresponde a uma função objetivo específica, e cada posição do vetor de autovalores também está relacionada a uma função objetivo, ou seja, a primeira posição é relacionada à primeira função objetivo. Esse passo corresponde as linhas 6, 7 e 8 do pseudocódigo 15.

Passo 3: Essa etapa compreende em analisar os autovetores e autovalores e identificar os objetivos conflitantes mais significativos, ou seja encontrar os objetivos que tem o maior valor por magnitude (maior valor absoluto). Esse passo corresponde as linhas 9, 10 e 11, do pseudocódigo 15. Para isso seguiremos uma sequencia de procedimentos:

- Definir um limiar(*threshold*), foi usado o mesmo que Saxena et al. (2013) 0.997, que é um número estatístico máximo que valor possui informações suficientes para ter significância, acima disso é considerado desprezível;
- Fazer uma soma gradativa das posições dos autovalores (posição 1 + posição 2 + posição 3...) até atingir o limiar de 0.997;
- Fazer uma contagem n da quantidade de autovalores que foram suficientes para atingir o limiar, e os n primeiros autovetores diretamente relacionados aos autovetores serão preservados, e os demais descartados.

Utilizando a matriz de autovetores cortada, será preservada posição referente ao objetivo mais significativo de cada autovetor. Esse passo corresponde no pseudocódigo 15, da linha 12 até a linha 21. Para isso será executado o seguinte processo:

- Percorrer cada autovetor e preservar o maior valor absoluto;
- Identificar de cada autovetor as posições que possuem sinais opostos ao maior valor absoluto e preservar também (por exemplo: se o maior valor absoluto for positivo, preserva todos os valores negativos);
- Se não existir valores opostos naquele autovetor, preserva o segundo maior valor absoluto.

No final de tudo, teremos o conjunto F_e que denota um conjunto de objetivos conflitantes $F_e \subseteq F$, onde se elimina de todos os posições dos autovetores, em que o objetivo não foi preservado por nenhum momento. Depois, os dados de entrada são reorganizados, considerando somente Fe. E então é novamente calculada a matriz de correlação (linear) RCM, ou de kernel (não-linear) K, a partir dos dados reduzidos. Passos correspondentes as linhas 22 e 23 do pseudocódigo 15.

Passo 4: Esse passo resume-se em analisar a matriz de correlação (linear) RCM ou a matriz kernel (não-linear) K, e identificar os objetivos potencialmente idênticos. Objetivos potencialmente idênticos são aqueles que cujo o valor seja maior que um *threshold* de correlação.

Esses procedimentos são correspondentes no pseudocódigo 15, da linha 24 até a linha 31. Segue a sequencia de passos que seguiremos:

- Inicialmente será calculado o *threshold* de correlação, denominado T_{cor} através da equação:

$$T_{cor} = 1 - e_1(1 - M_2\sigma/M) \tag{4.1}$$

  Onde e_1 representa o primeiro autovalor, M é a quantidade de autovalores e $M_2\sigma$ é definido pelo corte dos autovalores obtidos da seguinte forma:

  - Definir um limiar(ou*threshold*), foi usado o mesmo que Saxena et al. (2013) 0.954;
  - Fazer uma soma gradativa das posições dos autovalores (posição 1 + posição 2 + posição 3...) até atingir o limiar de 0.954;
  - Fazer uma contagem $M2\sigma$ da quantidade de autovalores que foram suficientes para atingir o limiar.

- Utilizando o valor de T_{cor}, será analisada a matriz (de correlação ou de kernel), para identificar os objetivos potencialmente idênticos, essa verificação é feita seguindo o seguinte processo:

  - Para cada item da matriz, a mesma será percorrida para encontrar a posição (x, y) cujo o valor seja maior que T_{cor}, caso seja, esse valor é considerado idêntico;
  - Por fim, subconjuntos serão criados contendo os objetivos idênticos entre si gerados na verificação.

Passo 5: Para finalizar, depois de identificar os subconjuntos de objetivos potencialmente idênticos, um *score* de cada objetivo será calculado, para definir qual objetivo será preservado e quais serão descartados. Consideramos que o objetivo do subconjunto que obtiver o melhor *score* será o escolhido a ser preservado e os demais serão desconsiderados. Esse passo é correspondente as linhas 31, 32 e 33 do pseudocódigo 15. A seguinte equação representa o cálculo do *score*:

$$score = \sum_{j=1}^{n_v} e_j \left| f_{ij} \right| \tag{4.2}$$

Onde para cada objetivo, é feito um somatório, da multiplicação do autovalor relacionado com todos os elementos do autovetor correspondente aquele objetivo.

A técnica não-linear para calculo da matriz também é baseada em análise de componentes principais (PCA), porém utilizando a extração não linear de recursos. Saxena et al. (2013) utilizou

uma técnica chamada NL-MVU-PCA (Nonlinear - *Maximum Variance Unfolding* - Análise de Componentes Principais, mas no sistema desenvolvido foi usado uma variedade de kernels.

O algoritmo trabalhado, usa uma função de kernel que define implicitamente o mapeamento não linear. Esse pode ser visto como uma generalização do PCA, em que as não-linearidades nos dados são levadas em conta quando a variação é maximizada, ou seja uma transformação não linear mapeia os dados de entrada em um espaço de característica dimensional mais alto(WIDJAJA et al., 2012).

Essa técnica analisa a matriz de Kernel K, calculada por vários tipos de kernels, para que seja feita uma redução objetiva não-linear. A implementação do algoritmo segue o mesmo fluxo do método linear, mas no método não-linear(NL-PCA) se utiliza o da matriz de kernel, e no método linear(L-PCA), utiliza a matriz de correlação.

Os kernels diferentes que foram utilizados para o cálculo da matriz de kernel K foram o Polinomial, o Gaussiano RBF (do inglês *Radial basis function*) e o Hiperbólico (Sigmoide). O cálculo da matriz kernel também foi feito com o auxilio da biblioteca WEKA, calculados através do dados iniciais. A motivação por trás da escolha de um kernel em particular pode ser muito intuitiva, dependendo do tipo de informação que esperamos extrair sobre os dados. Segundo Pierna et al. (2004) desde que algumas condições necessárias são satisfeitas, qualquer uma das muitas funções de kernel podem ser utilizadas nos diversos problemas. Como os kernels polinomial, RBF e sigmóide são bastantes encontrados na literatura com bons resultados, foram os escolhidos.

Kernel Polinomial: É um kernel que representa a similaridade de vetores em um espaço de característica sobre polinômios das variáveis originais e suas combinações, permitindo a aprendizagem de modelos lineares. O espaço de característica de um núcleo polinomial é equivalente ao da regressão polinomial, mas sem o aumento da combinações no número de parâmetros. A função Kernel é dada por:

$$k(x,y) = (\alpha x^T y + c)^d \tag{4.3}$$

Onde x e y são vetores no espaço de entrada, os parâmetros ajustáveis são o declive , o termo constante c e o grau polinomial d (KOKIOPOULOU; SAAD, 2004)

Kernel Gaussiano RBF: O kernel gaussiano RBF (*Radial basis function*) é uma função universal do kernel, e pode ser aplicado a qualquer uma das distribuições do amostras através da escolha de parâmetros. Tem sido mais usado no mapeamento não linear. Função do kernel RBF é dada pela função:

$$k(x,y) = e\left(-\frac{\|x-y\|^2}{2\sigma^2}\right) \tag{4.4}$$

Onde σ define a largura da função gaussiana e pode ser usado para ajustar o grau de generalização. Esse parâmetro ajustável σ desempenha um papel importante no desempenho do kernel e deve ser cuidadosamente ajustado para o problema em questão. Se superestimada, a exponencial se comportará quase linearmente e a projeção de maior dimensão começará a perder seu poder não linear. Por outro lado, se subestimada, a função não terá regularização e o limite de decisão será altamente sensível ao ruído nos dados de treinamento(HAN; QUBO; MENG, 2012; PIERNA et al., 2004).

Kernel Sigmoide: É um kernel da tangente hiperbólica, também conhecido como *Kernel Multilayer Perceptron* (MLP). O Kernel Sigmoide vem do campo Redes Neurais, onde a função sigmoide bipolar é muito usada como uma função de ativação para neurônios artificiais. Esse existem dois parâmetros ajustáveis no núcleo sigmoide, o $alpha$ do declive e a constante de interceptação c . Um valor comum para $alpha$ é $1/N$, onde N é a dimensão de dados. A equação desse kernel é expressa por (LIN; LIN, 2003):

$$k(x,y) = tanh(\alpha x^T y + c) \tag{4.5}$$

5
Experimentos

Este capítulo descreve os procedimentos para avaliação dos algoritmos de redução de objetivo propostos nesse trabalho. Os algoritmos são comparados a uma análise de *benchmark*. Como dados iniciais, foi usada uma base de dados gerada por Matos (2017). A partir desses dados, foram aplicados quatro algoritmos de redução de objetivos, o L-PCA, o K-PCA usando kernel Polinomial, o K-PCA com o kernel sigmóide e o K-PCA com o kernel RBF. Para cada algoritmo de redução, através do framework jMetal, dois algoritmos de otimização foram utilizados, o NSGA-II e o NSGA-III. Para esses algoritmos foram feitas análises, comparando os resultados dos algoritmos sem a redução de objetivos, com os algoritmos aplicando diferentes tipos de redução. Com isso foi possível avaliar a aplicação dos algoritmos de redução de objetivos se obtém melhoras no desempenho dos algoritmos de otimização.

A organização desse capítulo está no seguinte formato: seção 5.1 descreve o *benchmark* utilizado nos experimentos; seção 5.2 apresenta as medidas de qualidades que serão trabalhadas; mostra os algoritmos e os parâmetros utilizados; seção 5.4 mostra resultados dos indicadores de hipervolume e estatístico; e por fim na seção 5.5 são apresentados os resultados.

5.1 Metodologia

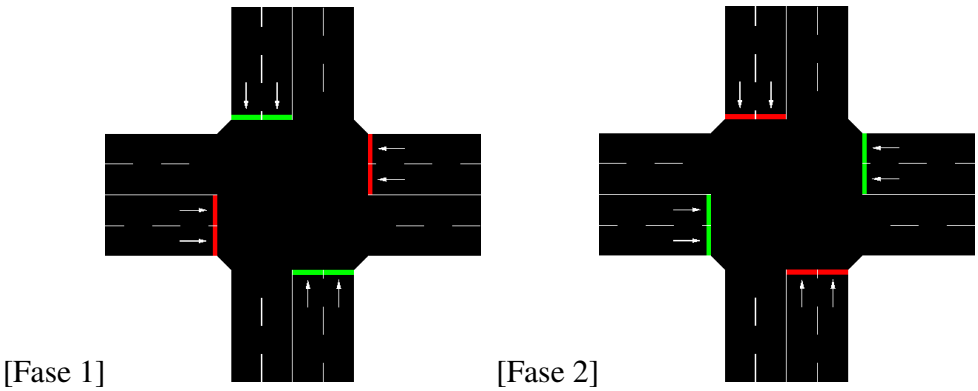
Nessa seção serão apresentados os passos da metodologia utilizada para realizar os experimentos. O primeiro passo consiste na aplicação dos métodos de redução de objetivos, onde através da base de dados, 4 algoritmos a procura de funções objetivo essenciais serão executados, L-PCA, K-PCA(Polinomial), K-PCA(RBF) e K-PCA(Sigmóide). O próximo passo compreende em executar uma otimização, utilizando os algoritmos NSGA-II e NSGA-III, no conjunto de dados iniciais contendo 12 objetivos, e nos subconjuntos oriundos dos algoritmos de redução (10 execuções para cada algoritmo, e para cada execução, 5060 avaliações(gerações)). Desse ponto, será feita a análise do desempenho de algoritmos de otimização, considerando somente os

objetivos reduzidos, análise feita através do cálculo de hipervolume das 10 execuções. Também serão feitas análises comparando o desempenho das otimizações dos algoritmos com os 12 objetivos, com os algoritmos de objetivos reduzidos. Depois, a melhor configuração de cada algoritmo de otimização será confrontada, buscando identificar qual algoritmo apresentar melhor desempenho na resolução do problema.

5.2 Benchmark

Para medir o desempenho, uma configuração de benchmark foi proposta. Essa configuração é composta por um cenário com 09 intersecções e 12 injetores para distribuição dos veículos, como indicados na figura 17. O fluxo definido foram utilizados 21.000 veículos, divididos por injetores que varia entre 900 a 2700 veículos, como indicado na tabela 1. A solução será codificada em um vetor de dezoito posições (n = 18), pois cada uma das 9 interseções possui duas fases "verdes".

No *benckmark* também são definidas o número de faixas que cada pista, e as fases semafóricas, como mostra na imagem16. Em todos os cruzamentos, contém 2 pistas de mão dupla, onde cada uma delas contém 4 faixas, sendo 2 faixas em cada sentido. As pistas são entendidas pelo simulador como arestas, e cada uma delas possui 188 metros de comprimento. Cada aresta tem um nó de origem e um de destino. Esses nós estão localizados nas extremidades do mapa, ou nas interseções. Cada interseção possui 12 metros de comprimento. Para o cenário utilizado, temos 48 arestas e 21 nós, sendo que em 9 desses temos os semáforos. Todas essas configurações foram mantidas em todos os experimentos.



[Fase 1] [Fase 2]

Figura 16 – Fases semafóricas

5.3 Medidas

Uma configuração importante do *benchmark* são as funções objetivos que serão saídas das simulações. Para todos os experimentos foram utilizadas 12 medidas de qualidades diferentes, 10 dessas são extraídas diretamente do SUMO (arquivo "tripinfo.xml", como na imagem **??**), e 2

Figura 17 – *Benchmark* do cenário com injetores

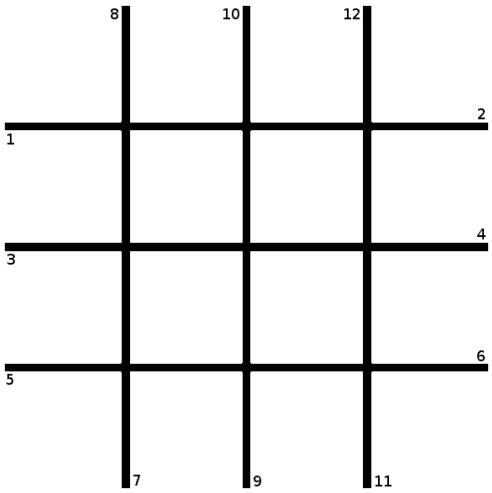


Tabela 1 – Quantidade de veículos por injetor

Injetor	Quantidade de Veículos
1	2100
2	2400
3	900
4	2700
5	1500
6	1100
7	1550
8	1200
9	2550
10	2260
11	990
12	1750

são obtidas através de cálculos baseados nas outras medidas. Essas medidas são obtidas dentre todos os veículos da simulação, por exemplo a primeira medida, *Depart delay*, é a soma dos valores de espera para começar a jornada de todos os veículos. Vale ressaltar que a comparação dos algoritmos envolve a análise de todas as doze medidas de qualidades, mesmo quando a otimização é feita a um subconjunto de medidas obtidas por uma redução. O experimento sempre verifica se 12 medidas são otimizadas para cada algoritmo. Seguem as medidas de qualidades utilizadas, enumeradas na tabela 2.

Para as comparações, os algoritmos são referenciados como apresentados na listagem a seguir:

NSGA-II: NSGA-II sem redução de objetivos;

NSGA-III: NSGA-III sem redução de objetivos;

+L-PCA: NSGA-II ou NSGA-III + o L-PCA;

Tabela 2 – Objetivos utilizados no trabalho

	Objetivo	**Descrição**
1	*Depart delay*	Tempo em segundos que o veículo esperou para começar sua jornada
2	*Trip duration*	Tempo em segundos necessário para o veículo realizar a rota
3	*Wait steps*	Número de etapas em que a velocidade do veículo ficou abaixo de 0.1 m/s
4	*Time loss*	Tempo perdido ao dirigir abaixo da velocidade ideal (60 km/h)
5	*CO abs*	Quantidade total em gramas de *CO* emitida pelo veículo durante a jornada
6	CO_2 *abs*	Quantidade total em gramas de CO_2 emitida pelo veículo
7	*Fuel abs*	Quantidade total de combustível utilizado pelo veículo durante a viagem
8	*HC abs*	Quantidade total de *HC* emitida pelo veículo
9	PM_x *abs*	Quantidade total de PM_x emitida pelo veículo
10	NO_x *abs*	Quantidade total de NO_x emitida pelo veículo durante a jornada
11	*Global mean speed*	Soma de todas as velocidades médias divididas pelo número de veículos
12	*Idle time*	A duração da viagem menos o tempo em que o veículo levaria para fazer a viagem sem paradas(tempo ideal)

+K-PCA(Pol): NSGA-II ou NSGA-III + o K-PCA usando o kernel Polinomial;

+K-PCA(RBF): NSGA-II ou NSGA-III + o K-PCA usando o kernel gaussiano RBF;

+K-PCA(Sig): NSGA-II ou NSGA-III + o K-PCA usando o kernel Sigmóide.

5.4 Algoritmos e Parâmetros

Nesta seção serão apresentados os parâmetros utilizados nos algoritmos de otimização (NSGA-II e NSGA-III), e os parâmetros dos algoritmo de redução de objetivos (L-PCA, K-PCA usando kernel gaussiano RBF, K-PCA usando kernel polinominal e K-PCA usando kernel sigmóide) utilizados nos experimentos. O processo de redução de objetivos é feito de forma *offline*, a partir da base de dados geradas por Matos (2017), contendo 12 objetivos. Os dados serão transformados em uma matriz e daí feito todo o processo de redução. Após esse processo, os algoritmos NSGA-II e NSGA-III serão executados 10 vezes para cada conjunto de objetivos preservados pela redução.

Os parâmetros dos algoritmos NSGA-II e NSGA-III utilizados na sua implementação, são definidos por padrão pelo *framework jMetal*, esse são: tamanho da população, tamanho individual, número de gerações, probabilidade de mutação e outros, apresentados na tabela 3. Para ser compreendido pelo jMetal, as soluções foram representadas como um vetor de repetições, em codificação binária, onde cada semáforo é representado por 7 bits (número de bits necessários para representar o maior valor que uma variável pode assumir nesse trabalho (1100100)). Para os algoritmos de redução, os parâmetros foram baseados no trabalho de Saxena et al. (2013), são eles: limiar de variância 1 (*threshold*) = 0,997, usado para identificar os objetivos conflitantes

Tabela 3 – Parâmetros do NSGA-II e NSGA-III

Parâmetro	**NSGA-II**	**NSGA-III**
Tamanho da população	100	92
Tamanho do indivíduo	18	18
Número de gerações	5060	5060
Probabilidade de mutação	1/7	1/7
Probabilidade de cruzamento	0.9	0.9
Índice de distribuição de mutação	20	20
Índice de distribuição de cruzamento	20	30
Método de mutação	*Bit flip*	*Bit flip*
Método de cruzamento	*Single point*	*Single point*
Operador de seleção	*Binary tournament*	*Binary tournament*

mais significativos, e limiar de variância 2 (*threshold*) = 0,954, usado para identificar os objetivos potencialmente idênticos.

5.5 Indicadores de Qualidade

Para medir a performance dos algoritmos multiobjetivo, são utilizados indicadores de qualidade. O hipervolume é uma medida utilizada para comparar algoritmos multiobjetivo. Essa utiliza uma medida maximização que determina a área atingida pela fronteira de Pareto aproximada. O teste de hipervolume é feito em cima de 10 execuções de cada algoritmo.

Para identificar a diferença estatística nas comparações dos algoritmos, melhor do NSGA-II e NSGA-III, é aplicado o teste de Wilcoxon, teste feito para comparar apenas a distribuição de duas amostras. E para comparar os algoritmos NSGA-II e NSGA-III com os quatro algoritmos de redução, é aplicado uma extensão do teste de Wilcoxon, o pós-teste de Kruskal-wallis, que é um método não paramétrico para testar se amostras se originam da mesma distribuição. Utilizando um nível de significância de 5%, os testes fazem analises dos hipervolumes das 10 execuções dos algoritmos, e comparações para identificar se os algoritmos possuem uma diferença estatística entre si.

5.6 Resultados

Nessa seção serão apresentados os detalhes dos experimentos realizados e serão discutidos os resultados. Todos os experimentos foram executados com as configurações do cenário descritas no *benckmark*. Através da base de dados, 4 algoritmos de redução de objetivos a procura de funções objetivo essenciais foram executados, L-PCA, K-PCA(Polinomial), K-PCA(RBF) e K-PCA(Sigmóide). Para cada conjunto de objetivos reduzidos gerados serão otimizados pelos algoritmos NSGA-II e NSGA-III, realizando 10 execuções para cada algoritmo. Para cada execução foram feitas 5060 avaliações(gerações).

5.6.1 Análise da redução

Os algoritmos de redução de objetivos procuram por funções objetivos essenciais, através dos objetivos correlacionados. Para o algoritmo linear L-PCA, foram encontrados 5 subconjuntos de funções objetivas correlacionados $\{1\},\{2,3,4,11,12\},\{3,5,7\},\{6,9,10\}$ e $\{8\}$, e como foi visto na seção anterior, para definir qual objetivo será preservado, uma escolha foi feita através do cálculo do *score* de cada objetivo, preservando assim o objetivo de maior *score*. Essa escolha é feita da seguinte forma: considerando o subconjunto $\{3,5,7\}$, os *score* são respectivamente 0.2324904710400307, 0.1210052717944119 e 0.0009482460887371, onde é preservado o primeiro objetivo (de número 3), cujo o *score* é maior. Feito isso em todos os subconjuntos, o L-PCA encontrou 4 objetivos essenciais, 1, 3, 6 e 8, são eles a seguir:

1 - Depart delay;

3 - Wait steps;

6 - CO_2 abs;

8 - HC abs.

Na busca por funções objetivos essenciais, o algoritmo linear K-PCA, utilizando o kernel Polinomial, encontrou 5 subconjuntos de funções objetivo correlacionados $\{1\}$,$\{2,3,11\}$, $\{3,5,7,12\}$, $\{4,6,9,10\}$ e $\{8\}$, e como foi visto para o L-PCA, para definir qual objetivo será preservado, uma escolha foi feita através do cálculo do *score* de cada objetivo, preservando assim o objetivo de maior *score*. Feito todo o procedimento em todos os subconjuntos, o K-PCA com kernel polinomial encontrou 5 objetivos essenciais, 1, 3, 5, 6 e 8 são eles:

***1 - Depart delay*;**

***3 - Wait steps*;**

***5 - CO abs*;**

***6 - CO_2 abs*;**

***8 - HC abs*.**

Para o K-PCA, utilizando o kernel Sigmóide, seguindo o mesmo procedimento listado anteriormente, foi encontrado 5 subconjuntos de funções objetivos correlacionados, $\{1\}$,$\{2,3,4\}$, $\{3,5,7,12\}$, $\{6,9,10,11\}$ e $\{8\}$, obtendo por fim 5 objetivos essenciais, são eles: 1, 2, 3, 6 e 8 listados a seguir:

***1 - Depart delay*;**

***2 - Trip duration*;**

***3 - Wait steps*;**

***6 - CO_2 abs*;**

***8 - HC abs*.**

Continuando com o mesmo processo, para o K-PCA, utilizando o kernel Gaussiano RBF, os subconjuntos de objetivos correlacionados encontrados foram os seguintes, $\{1\}$,$\{2,4,9\}$, $\{2,3,5,7\}$, $\{6,9,10,11,12\}$ e $\{8\}$, obtiveram 5 objetivos preservados, são eles: 1, 2, 5, 6 e 8 enumerados a seguir:

***1 - Depart delay*;**

***2 - Trip duration*;**

***5 - CO abs*;**

***6 - CO_2 abs*;**

***8 - HC abs*.**

Para todos os algoritmos foi obtido uma considerável redução de objetivos. Para o algoritmo linear foi reduzido 41.66% e para os algoritmos não-lineares 33.33%. As funções objetivo selecionadas pelos algoritmos são diferentes. A funções objetivos selecionadas pelos algoritmos estão compreendidadas entre 1,2,3,5,6 e 8. As funções 1, 6 e 8 foram comuns em todos os algoritmos. A função objetivo de número 3, só não foi selecionada no algoritmo K-PCA usando o kernel gaussiano RBF. E a função objetivo de número 2, foi apenas selecionada pelo algoritmo K-PCA usando kernel sigmóide e K-PCA usando kernel gaussiano RBF.

Apesar dos algoritmos identificarem objetivos conflitantes, não houve unanimidade na escolha. Uma comparação entre os algoritmos é importante para identificar o efeito da redução na qualidade das soluções geradas pelos algoritmos.

5.6.2 Comparação de algoritmos

Considerando os conjuntos dos objetivos reduzidos encontrados pelos algoritmos, o NSGA-II e o NSGA-III foram executados 10 vezes. Para realizar uma comparação, foi usado um indicador de qualidade, denominado hipervolume. Onde basicamente, quanto maior o valor do hipervolume, melhor é a qualidade das soluções, ou seja o algoritmo que obtiver os melhores hipervolumes é considerado a melhor solução. Uma comparação foi feita entre as soluções de otimização dos algoritmos NSGA-II e NSGA-III, das execuções sem a redução de objetivos, com os resultados com a utilização dos algoritmos de redução objetiva L-PCA e K-PCA com

diferentes kernels. Por fim são comparados os algoritmos NSGA-II e NSGA-III, buscando identificar qual combinação de algoritmo e técnica de redução obtém o melhor resultado para o problema.

O hipervolume foi calculado para cada execução para todos os algoritmos, gerando 10 valores de por algoritmo. A Tabela 4 apresenta a média, desvio padrão, valores máximo e mínimo do hipervolume para cada algoritmo usando o NSGA-II e a tabela 5 apresenta a média, desvio padrão, valores máximo e mínimo do hipervolume para cada algoritmo usando o NSGA-III. As figura 18 e 19 apresenta os *boxplots* dos hipervolumes obtidos.

Também foi realizada uma comparação estatística utilizando o teste de Kruskal-wallis, onde através dos 10 valores de hipervolumes gerados de cada algoritmo, são comparados. A tabela usando o NSGA-II 6 mostra que nas comparações não houve diferença estatística em nenhum dos algoritmos, pois o valor de $p - value > 0.05$. Para o algoritmo NSGA-III, a tabela 7, mostra que nas comparações não houve diferença estatística entre os algoritmos, pois $p - value > 0.05$.

Para o NSGA-II, o valor do hipervolume máximo atingido foi 0.8739625, comparando aos outros algoritmos +K-PCA(Poli), +K-PCA(RBF), +K-PCA, +L-PCA, obtiverem os valores respectivamente, 0.8875577, 0.8308641, 0.8553707 e 0.7941471, observa-se que não houve uma diferença tão grande, o mesmo acontece com o valor mínimo atingido, para o NSGA-II foi 0.4894049 e para os outros algoritmos respectivamente 0.5896125, 0.6001665, 0.5678744 e 0.5052838, é nítido que não há grande diferença. O desvio padrão entre os algoritmos também são valores baixos, 0.1105874, 0.1011352, 0.08038063, 0.09732771 e 0.09881543, ou seja, não houve muita dispersão entre os valores. Analisando agora o NSGA-III, os valores houve uma variação um pouco maior que no NSGA-II, mas não uma diferença considerável. Podemos observar os valores nos máximos dos hipervolumes, para o NSGA-III obteve 1.942035, e para os algoritmos de redução +K-PCA(Poli), +K-PCA(RBF), +K-PCA e +L-PCA, na devida ordem, 1.778945, 0.7241665, 0.8293965 e 0.8053068. Já os mínimos atingidos por cada algoritmo foram bem próximos, para o NSGA-III e todos os algoritmos de redução de modo respectivo, 0.4939384, 0.2165641, 0.4536884, 0.4448561 e 0.4427838. E para desvio padrão, houve uma dispersão um pouco maior que do NSGA-II, mas ainda assim foram valores baixos, observamos respectivamente os valores NSGA-III 0.4257191, e os demais com redução 0.4380685, 0.08468891, 0.1406824 e 0.1091977.

Analinado as médias dos hipervolumes, para o NSGA-II pode-se observar que um dos algoritmos de redução conseguiu superar os hipervolumes das execuções sem redução, o algoritmo de redução K-PCA, usando o kernel polinomial, conseguiu obter hipervolumes superiores ao do NSGA-II sem redução. Apesar dos demais algoritmos não terem superados, os valores médios dos hipervolumes foram bem parecidos e também não houve diferença estatística entre si. O algoritmo que obteve os hipervolumes mais baixos foi o L-PCA. Para o NSGA-III, pode-se observar que nenhum dos algoritmos de redução conseguiu superar os hipervolumes

Tabela 4 – Tabela de hypervolule do NSGA-II

Hyper-volume	**NSGA-II**	**+K-PCA(Poli)**	**+K-PCA(RBF)**	**+K-PCA(Sig)**	**+L-PCA**
Média	0.7339345	0.7646251	0.7119148	0.7174371	0.6541216
Desvio Padrão	0.1105874	0.1011352	0.08038063	0.09732771	0.09881543
Máximo	0.8739625	0.8875577	0.8308641	0.8553707	0.7941471
Mínimo	0.4894049	0.5896125	0.6001665	0.5678744	0.5052838

Tabela 5 – Tabela de hypervolule do NSGA-III

Hyper-volume	**NSGA-III**	**+K-PCA(Poli)**	**+K-PCA(RBF)**	**+K-PCA(Sig)**	**+L-PCA**
Média	0.8281815	0.6316272	0.5469646	0.6159282	0.5796779
Desvio Padrão	0.4257191	0.4380685	0.08468891	0.1406824	0.1091977
Máximo	1.942035	1.778945	0.7241665	0.8293965	0.8053068
Mínimo	0.4939384	0.2165641	0.4536884	0.4448561	0.4427838

Tabela 6 – Tabela do teste estatístico do NSGA-II

p-value	**NSGA-II**	**+K-PCA(Poli)**	**+K-PCA(RBF)**	**+K-PCA(Sig)**	**+L-PCA**
NSGA-II	-	0.99	0.97	0.99	0.32
+K-PCA(Poli)	-	-	0.81	0.86	0.13
+K-PCA(RBF)	-	-	-	1.00	0.71
+K-PCA(Sig)	-	-	-	-	0.64
+L-PCA	-	-	-	-	-

das execuções sem redução. Apesar de não superar, o algoritmo de redução que obteve melhor resultado foi utilizando o K-PCA, usando o kernel polinomial. Mesmo não superando o NSGA-III sem redução, os valores médios dos hipervolumes foram bem próximos, e também não houve diferença estatística.

E o teste estatístico desses algoritmos indicam que não houve diferença estatística. Ou seja, os resultados foram significativamente equivalentes, mesmo quando não obteve melhoras nos hipervolumes, podendo afirmar que as técnicas de redução influenciaram a busca levando ao algoritmo ter diferentes soluções. Assim podendo trabalhar com um subconjunto com um terço do tamanho do conjunto original e obter resultados com qualidades equiparadas.

Além disso, foi feita uma comparação estatística utilizando o teste de Wilcoxon, onde através dos 10 valores de hipervolumes gerados pelos algoritmo do NSGA-II e NSGA-III sem redução, e pode-se observar na imagem 20 que o NSGA-III obteve o melhor resultado.

Tabela 7 – Tabela do teste estatístico do NSGA-III

p-value	**NSGA-III**	**+K-PCA(Poli)**	**+K-PCA(RBF)**	**+K-PCA(Sig)**	**+L-PCA**
NSGA-III	-	0.18	0.17	0.69	0.42
+K-PCA(Poli)	-	-	1.00	0.90	0.99
+K-PCA(RBF)	-	-	-	0.89	0.99
+K-PCA(Sig)	-	-	-	-	0.99
+L-PCA	-	-	-	-	-

Tabela 8 – Tabela do teste estatístico do NSGA-II e NSGA-III

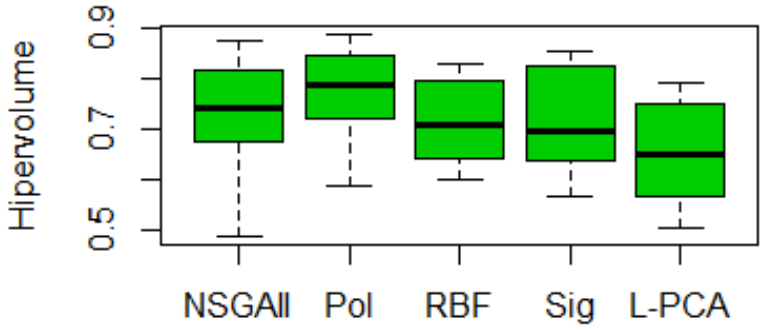
p-value	NSGA-II	NSGA-III
NSGA-II	-	0.94
NSGA-III	-	-

Tabela 9 – Tabela da média das medidas de qualidade das melhores soluções

	NSGA-III	**KPCA(Pol)**	**KPCA(RBF)**	**KPCA(Sig)**	**L-PCA**
Obj. 01	0.140511768	0.097698613	0.076112061	0.090658561	0.064832534
Obj. 02	0.155190099	0.208240916	0.124653567	0.136706986	0.137012598
Obj. 03	0.098133411	0.118571638	0.042538081	0.066918101	0.058495496
Obj. 04	0.155167221	0.208205434	0.124635907	0.136668588	0.136985314
Obj. 05	0.091118702	0.111800469	0.046877890	0.058648469	0.040335242
Obj. 06	0.081852650	0.087843301	0.036598109	0.041784834	0.033802439
Obj. 07	0.090519894	0.109772587	0.046029074	0.057398113	0.039635543
Obj. 08	0.102918175	0.092024353	0.049662862	0.059041559	0.041232293
Obj. 09	0.083365634	0.086247968	0.036840957	0.042025960	0.034240091
Obj. 10	0.081852441	0.087846365	0.036598595	0.041786009	0.033802684
Obj. 11	0.665669856	0.733128159	0.649316320	0.683231487	0.705335312
Obj. 12	0.155190099	0.208240916	0.124653567	0.136706986	0.137012598

Um resultado esperado, pois segundo a literatura, o NSGA-III é mais adequado para resolver problemas com muitos objetivos. Através da tabela 8, podemos observar também que não houve diferença estatística entre os algoritmos, pois $p - value$ 0.94 > 0,05, ou seja, mesmo obtendo resultados diferentes, todos são capazes de trabalhar com o problema com uma boa qualidade.

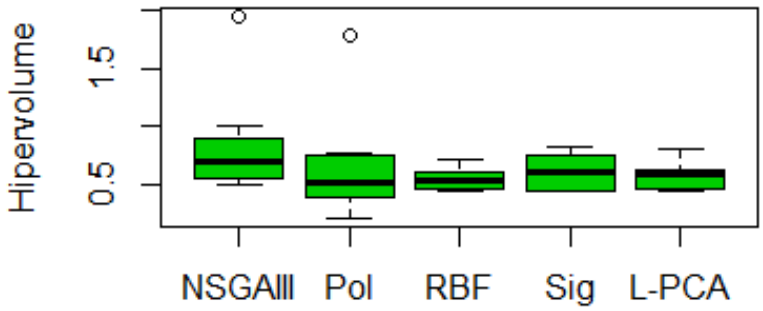
Figura 18 – *Boxplot* do hipervolume para cada algoritmo usando NSGA-II



Fonte: O autor

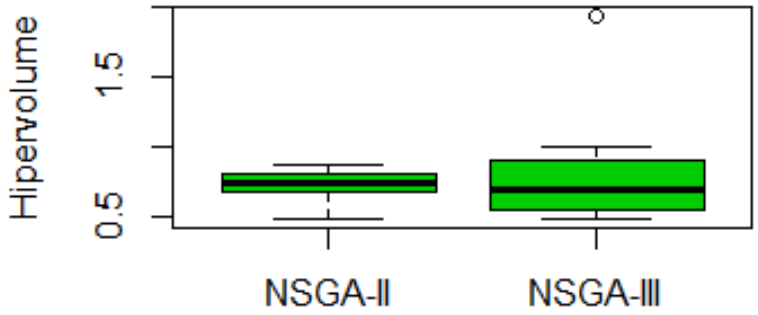
Como o algoritmo NSGA-III obteve melhores resultados, foi feita uma analise de dados

Figura 19 – *Boxplot* do hipervolume para cada algoritmo usando NSGA-III



Fonte: O autor

Figura 20 – *Boxplot* do hipervolume dos algoritmos NSGA-II e NSGA-III



Fonte: O autor

das medidas de qualidades. De cada algoritmo, foi considerado as soluções da execução que obteve o melhor hipervolume (considerando que cada algoritmo teve 10 execuções em busca de melhores soluçoes).

A média dos valores de cada objetivo, dentre as soluções obtidas por cada algoritmo foram calculadas e expostas na tabela 9, e gerado o gráfico 21. Fazendo uma comparação entre o algoritmo NSGA-III (sem redução de objetivos) e os demais algoritmos com a redução de

objetivos, podemos observar que não tem uma diferença considerável nos valores para cada objetivo. Se considerar o objetivo 1, as médias a seguir são: NSGA-III = 0.14051176884; KPCA(Pol) = 0.09769861308; KPCA(RBF)= 0.07611206161; KPCA(Sig) = 0.09065856120; L-PCA = 0.06483253440 a menor média tem uma diferença menor que 0.08, um valor considerado baixo, ou seja, mesmo utilizando os métodos de redução considerando uma menor quantidade de objetivos, obtém soluções equivalentes.

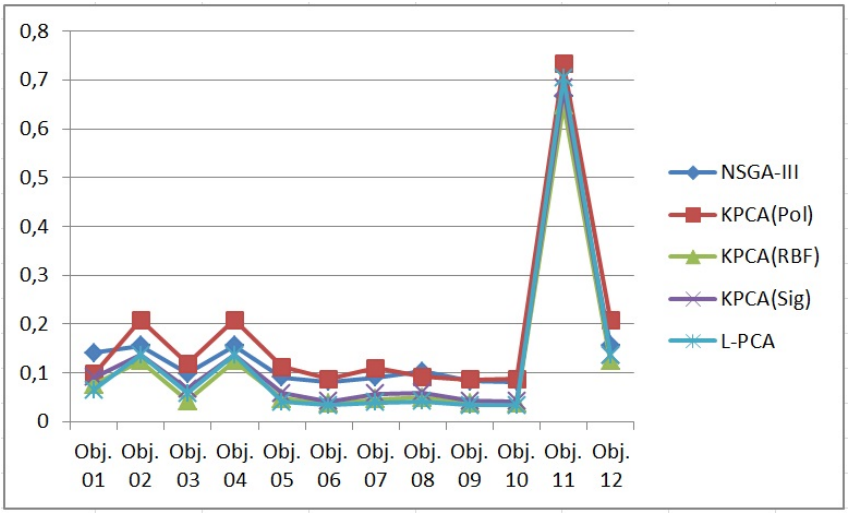
Fazendo uma análise mais refinada dos algoritmos e objetivos, os gráficos 22, 23, 24, 25 e 26 mostram os valores que cada medida de qualidade representam, da melhor solução daquele algoritmo. No eixo x, está presentado pela numeração dos objetivos de 1 ao 12; no eixo y, está a escala numérica que os valores das medidas de qualidade atingem, compreendidos entre 0 e 1. Cada linha representa uma solução, e os pontos representam os valores para aquele determinado objetivo. Mesmo na representação de um algoritmo com redução, que trabalha com menos objetivos, a representação é feita para todos os objetivos. Pode se observar nos formatos das linhas, que existem uma similaridade, ou seja, as soluções são tem valores bem próximos.

Podemos observar também, que os algoritmos de redução, mesmo considerando somente alguns objetivos, todos objetivos também são otimizados, como por exemplo o gráfico 23, do algoritmo +K-PCA(Pol), que trabalha com o subconjunto dos objetivos 1, 3, 5, 6 e 8, porém nota-se que todos os objetivos tem valores baixos, e não somente os trabalhados. Esse detalhe acontece em todos os algoritmos. Observa-se também que o objetivo 11, houve uma dificuldade de otimização em todos os casos, nota-se que apenas duas soluções conseguiram obter uma melhora, representado nos gráficos 22 e 23.

Uma característica que se observa nos gráficos, é que no algoritmo NSGA-III sem redução, é comum que se obtenha mais soluções, como pode ser visto no gráfico 22. Já para os algoritmos reduzidos, a quantidade de soluções ótimas diminuem, como mostra os gráficos (23, 24, 25 e 26), que tem menos soluções, em alguns casos para uma execução só obteve apenas 1 ou 2 melhores soluções. O algoritmo K-PCA utilizando o kernel polinomial, é uma excessão, pois obteve uma quantidade grande de soluções boas por execução. Se analisar gráfico 23 do K-PCA(Pol), pode-se observar que existem muitas soluções comparados aos outros algoritmos de redução, que obtiveram apenas 2 ou 3 melhores soluções.

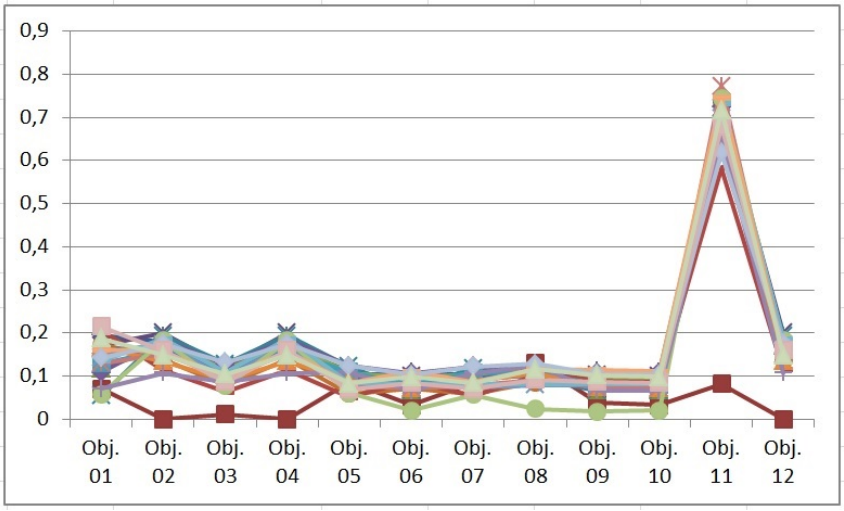
Os experimentos trabalhados foram os seguintes, a otimização abrangendo 12 objetivos com o uso do NSGA-II e do NSGA-III, e também a otimização dos subconjuntos obtidos pelos algoritmos de redução L-PCA, K-PCA(Pol), K-PCA(RBF) e K-PCA(Sig), todos usando os algoritmos NSGA-II e NSGA-III. Concluiu que a otimização do NSGA-III sobre o K-PCA(Pol) obteve resultados superiores ao NSGA-III sem redução. Para o NSGA-II, nenhuma otimização com objetivos reduzidos conseguiu resultados melhores que o algoritmo sem redução, porem o algoritmo que mais se aproximou foi o K-PCA(Pol), concluindo que o K-PCA(Pol) obteve uma melhor performance tanto no NSGA-II e NSGA-III. Uma comparação entre o NSGA-II e NSGA-III foi realizada, e o NSGA-III superou nos resultados. Vale ressaltar também que ao

Figura 21 – Média das medidas de qualidade das melhores soluções de cada algoritmo



Fonte: O autor

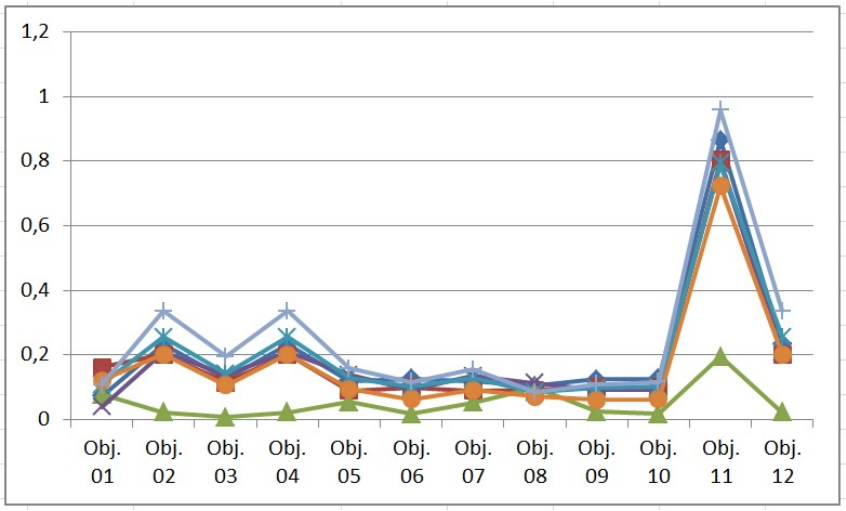
Figura 22 – Medidas de qualidade das melhores soluções do NSGA-III



Fonte: O autor

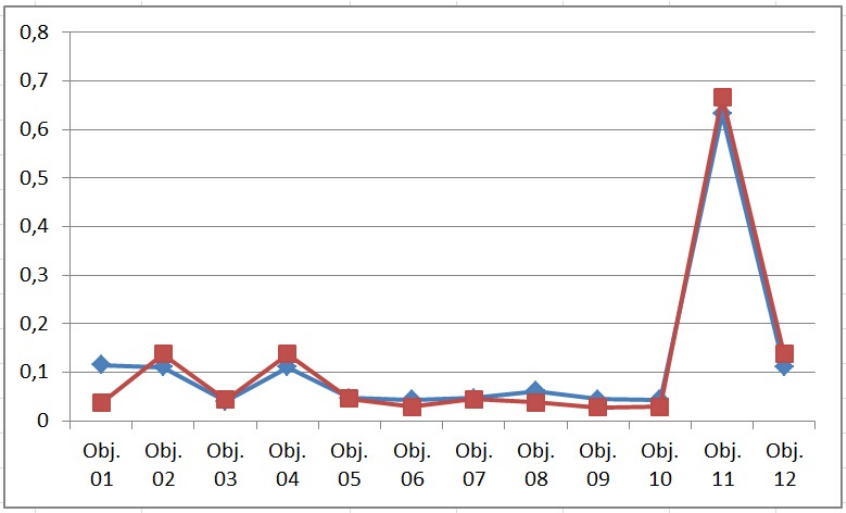
fazer o teste de diferença estatística, em nenhum algoritmo houve uma diferença entre o outro.

Figura 23 – Medidas de qualidade das melhores soluções do NSGA-III + KPCA(Pol)



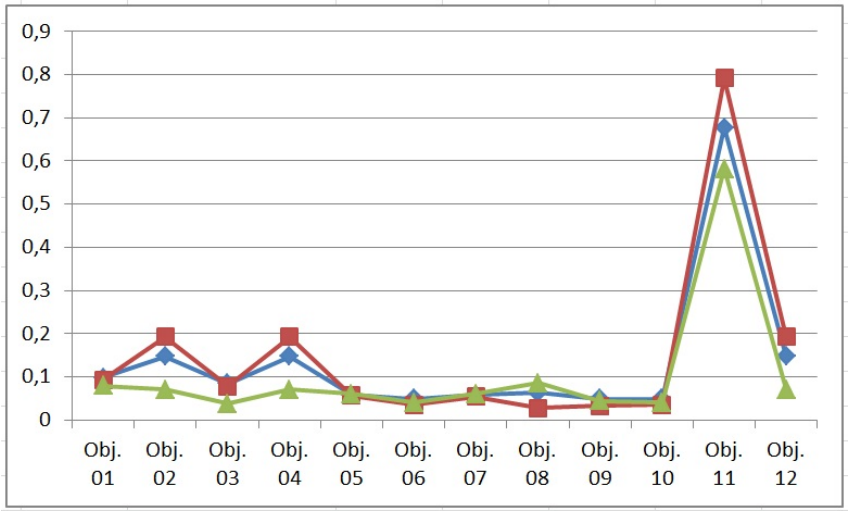
Fonte: O autor

Figura 24 – Medidas de qualidade das melhores soluções do NSGA-III + KPCA(RBF)



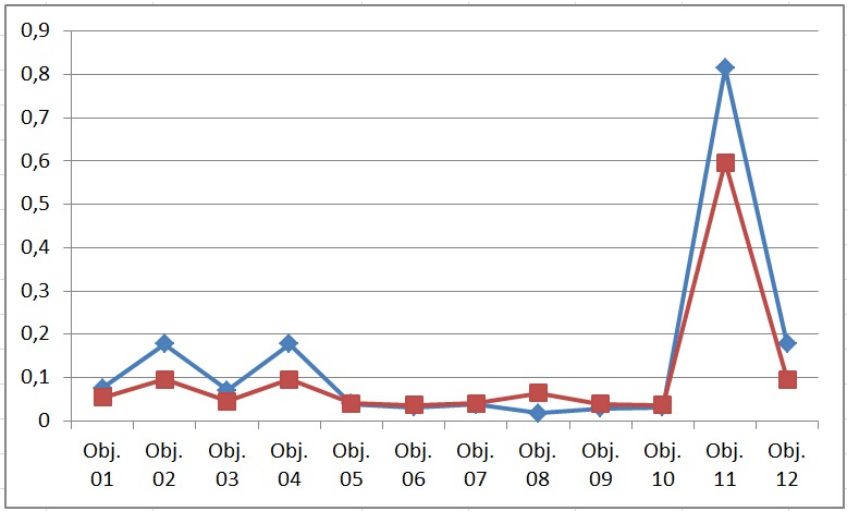
Fonte: O autor

Figura 25 – Medidas de qualidade das melhores soluções do NSGA-III + KPCA(Sig)



Fonte: O autor

Figura 26 – Medidas de qualidade das melhores soluções do NSGA-III + LPCA



Fonte: O autor

6

Conclusão

A principal contribuição de realizar este estudo, parte da premissa de mostrar que é possível utilizar técnicas de aprendizagem de máquina para redução de objetivos, nos problemas de otimização de sincronização de semáforos e obter bons resultados. Para o estudo, foi utilizada uma base de dados como entrada, contendo doze funções objetivo. Os algoritmos L-PCA, K-PCA com três kernels diferentes (Polinomial, RBF e Sigmóide), foram utilizados para reduzir a quantidade de objetivos. Com o uso do L-PCA, foi possível reduzir de doze objetivos para apenas quatro, e com o K-PCA a redução foi de doze objetivos para cinco. Os objetivos trabalhados foram *Depart delay, Trip duration, Wait steps, Time loss, CO abs, CO_2 abs, Fuel abs, HC abs, PM_x abs, NO_x abs, Global mean speed e Idle time*. Todos os experimentos foram aplicados a uma rede semafórica com 9 cruzamentos, a um fluxo moderado. Para construir esse cenário, foi utilizado um simulador de tráfego(SUMO) para criar representação computacional e se extrair as medidas de qualidades que foram otimizadas.

Após a redução de objetivos, baseada no sistema de redução de dimensionalidade proposto por Saxena et al. (2013), foi aplicada uma otimização, onde foi usados os algoritmos NSGA-II e NSGA-III. O algoritmo NSGA-III se destacou devido aos melhores resultados encontrados nas comparações de hipervolume. Comparações foram feitas entre a otimização feita pelos algoritmos NSGA-II e NSGA-III sem a aplicação de uma redução de objetivos, e das execuções de otimização com a utilização do L-PCA e K-PCA para a redução de objetivos. Analisando foi possível notar que somente no NSGA-III, um algoritmo de redução superou os resultados, o K-PCA usando kernel Polinomial, a otimização com os objetivos reduzidos conseguiu superar. Além disso, mostrou que mesmo não superando nos outros casos, os algoritmos de redução obtiveram resultados bem semelhantes, ou seja, mostrando que é possível trabalhar o problema com uma menor quantidade de objetivos e conseguir a mesma qualidade nos resultados. Assim, conclui-se que o uso da técnica de redução de objetivo foi muito eficaz no problema de sincronização de semáforos.

Para trabalhos futuros, é proposto a implementação de mais técnicas de redução de objetivos ainda não exploradas no contexto de Otimização com Muitos Objetivos para o o problema de sincronização de semáforos. Também, executar um conjunto maior de experimentos contendo cenários diferentes com mais variações de fluxos, para melhor avaliar os algoritmos propostos. Além disso, criar uma aplicação automática para executar algoritmos de otimização, aplicar técnicas de redução de objetivos e reexecutar os algoritmos com os objetivos reduzidos, tudo de forma *online*.

Referências

ABDOOS, M.; MOZAYANI, N.; BAZZAN, A. L. Traffic light control in non-stationary environments basewd on multi agent q-learning. In: IEEE. *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*. [S.l.], 2011. p. 1580–1585. Citado 2 vezes nas páginas 11 e 29.

BROCKHOFF, D. *Java Implementations of Exact and Greedy Objective Reduction Algorithms*. [S.l.: s.n.], 2006. Citado 2 vezes nas páginas 12 e 17.

BROCKHOFF, D.; ZITZLER, E. Are all objectives necessary? on dimensionality reduction in evolutionary multiobjective optimization. In: *Parallel Problem Solving from Nature-PPSN IX*. [S.l.]: Springer, 2006. p. 533–542. Citado na página 18.

BROCKHOFF, D.; ZITZLER, E. Objective reduction in evolutionary multiobjective optimization: Theory and applications. *Evolutionary computation*, MIT Press, v. 17, n. 2, p. 135–166, 2009. Citado na página 18.

CARVALHO, A. B. d. Novas estratégias para otimização por nuvem de partículas aplicadas a problemas com muitos objetivos. 2013. Citado 3 vezes nas páginas 12, 15 e 17.

COELLO, C. A. C. et al. *Evolutionary algorithms for solving multi-objective problems*. [S.l.]: Springer, 2007. v. 5. Citado na página 15.

DEB, K. et al. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In: SPRINGER. *International Conference on Parallel Problem Solving From Nature*. [S.l.], 2000. p. 849–858. Citado 3 vezes nas páginas 19, 20 e 21.

DEB, K.; JAIN, H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Trans. Evolutionary Computation*, v. 18, n. 4, p. 577–601, 2014. Citado 3 vezes nas páginas 17, 21 e 22.

DEB, K.; JAIN, H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Trans. Evolutionary Computation*, v. 18, n. 4, p. 577–601, 2014. Citado na página 21.

DEB, K.; SUNDAR, J. Reference point based multi-objective optimization using evolutionary algorithms. In: ACM. *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. [S.l.], 2006. p. 635–642. Citado na página 18.

GARCIA-NIETO, J.; OLIVERA, A. C.; ALBA, E. Optimal cycle program of traffic lights with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 17, n. 6, p. 823–839, 2013. Citado na página 29.

GRIGGS, W. M. et al. A large-scale sumo-based emulation platform. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 16, n. 6, p. 3050–3059, 2015. Citado na página 31.

HAJBABAIE, A.; BENEKOHAL, R. F. A program for simultaneous network signal timing optimization and traffic assignment. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 16, n. 5, p. 2573–2586, 2015. Citado 2 vezes nas páginas 19 e 30.

HAN, S.; QUBO, C.; MENG, H. Parameter selection in svm with rbf kernel function. In: IEEE. *World Automation Congress (WAC), 2012*. [S.l.], 2012. p. 1–4. Citado na página 42.

HANCKE, G. P.; JR, G. P. H. et al. The role of advanced sensing in smart cities. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 13, n. 1, p. 393–425, 2012. Citado na página 27.

HERRERA-QUINTERO, L. F. et al. Smart cities approach for colombian context. learning from its experiences and linking with government organization. In: IEEE. *Smart Cities Symposium Prague (SCSP), 2015*. [S.l.], 2015. p. 1–6. Citado na página 28.

JORGE, C. A. C. et al. Algoritmo evolutivo multi-objetivo de tabelas para seleção de variáveis em calibração multivariada. Universidade Federal de Goiás, 2014. Citado na página 20.

KHAMIS, M. A.; GOMAA, W. Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 29, p. 134–151, 2014. Citado 4 vezes nas páginas 11, 12, 22 e 31.

KOKIOPOULOU, E.; SAAD, Y. Pca and kernel pca using polynomial filtering: a case study on face recognition. *University of Minnesota*, 2004. Citado na página 41.

KRAJZEWICZ, C.; RÖSSEL, D. *SUMO User Documentation Rev. 1.24*. 2006. Citado na página 31.

KWASNICKA, H.; STANEK, M. Genetic approach to optimize traffic flow by timing plan manipulation. In: IEEE. *Intelligent Systems Design and Applications, 2006. ISDA'06. Sixth International Conference on*. [S.l.], 2006. v. 2, p. 1171–1176. Citado 2 vezes nas páginas 12 e 23.

LIN, H.-T.; LIN, C.-J. A study on sigmoid kernels for svm and the training of non-psd kernels by smo-type methods. *submitted to Neural Computation*, v. 3, p. 1–32, 2003. Citado na página 42.

MATOS, S. A. Sincronizacao de semaforos como um problema de otimizacao com muitos objetivos. *Dissertação - Universidade Federal de Sergipe*, 2017. Citado 7 vezes nas páginas 11, 12, 23, 26, 30, 43 e 46.

MCKENNEY, D.; WHITE, T. Distributed and adaptive traffic signal control within a realistic traffic simulation. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 26, n. 1, p. 574–583, 2013. Citado 2 vezes nas páginas 11 e 27.

MIKA, S. et al. Kernel pca and de-noising in feature spaces. In: *Advances in neural information processing systems*. [S.l.: s.n.], 1999. p. 536–542. Citado na página 64.

OLIVEIRA, D. d. Um estudo de coordenação dinâmica de agentes aplicado ao gerenciamento de tráfego veicular urbano. 2005. Citado na página 29.

OLIVEIRA, D. D.; BAZZAN, A. L. Traffic lights control with adaptive group formation based on swarm intelligence. In: SPRINGER. *International Workshop on Ant Colony Optimization and Swarm Intelligence*. [S.l.], 2006. p. 520–521. Citado na página 11.

PENG, W.; JIANG-PING, W.; JING, X. The application of particle swarm optimization on intelligent transport system. In: IEEE. *Computing, Communication, Control, and Management, 2009. CCCM 2009. ISECS International Colloquium on*. [S.l.], 2009. v. 4, p. 389–391. Citado na página 29.

PIERNA, J. F. et al. Combination of support vector machines (svm) and near-infrared (nir) imaging spectroscopy for the detection of meat and bone meal (mbm) in compound feeds. *Journal of Chemometrics: A Journal of the Chemometrics Society*, Wiley Online Library, v. 18, n. 7-8, p. 341–349, 2004. Citado 2 vezes nas páginas 41 e 42.

SATO, H.; AGUIRRE, H. E.; TANAKA, K. Controlling dominance area of solutions and its impact on the performance of moeas. In: SPRINGER. *International conference on evolutionary multi-criterion optimization*. [S.l.], 2007. p. 5–20. Citado na página 18.

SAXENA, D. K. et al. Objective reduction in many-objective optimization: Linear and nonlinear algorithms. *IEEE Transactions on Evolutionary Computation*, IEEE, v. 17, n. 1, p. 77–99, 2013. Citado 10 vezes nas páginas 12, 13, 17, 18, 35, 36, 39, 40, 46 e 58.

SEREDYNSKI, M.; MAZURCZYK, W.; KHADRAOUI, D. Multi-segment green light optimal speed advisory. In: IEEE. *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2013 IEEE 27th International*. [S.l.], 2013. p. 459–465. Citado na página 29.

SHEN, Z.; WANG, K.; WANG, F.-Y. Gpu based non-dominated sorting genetic algorithm-ii for multi-objective traffic light signaling optimization with agent based modeling. In: IEEE. *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*. [S.l.], 2013. p. 1840–1845. Citado 2 vezes nas páginas 12 e 19.

STEVANOVIC, A. et al. Multi-criteria optimization of traffic signals: Mobility, safety, and environment. *Transportation Research Part C: Emerging Technologies*, Elsevier, v. 55, p. 46–68, 2015. Citado na página 22.

SUN, D.; BENEKOHAL, R. F.; WALLER, S. T. Multiobjective traffic signal timing optimization using non-dominated sorting genetic algorithm. In: IEEE. *Intelligent vehicles symposium, 2003. proceedings. ieee*. [S.l.], 2003. p. 198–203. Citado 2 vezes nas páginas 12 e 22.

VIEIRA, J. et al. Objective reduction on many-objective traffic lights signaling optimization. *IEEE Symposium on Computers and Communications*, 2018. Citado na página 14.

WIDJAJA, D. et al. Application of kernel principal component analysis for single-lead-ecg-derived respiration. *IEEE Transactions on Biomedical Engineering*, IEEE, v. 59, n. 4, p. 1169–1176, 2012. Citado na página 41.

WOLD, S.; ESBENSEN, K.; GELADI, P. Principal component analysis. *Chemometrics and intelligent laboratory systems*, Elsevier, v. 2, n. 1-3, p. 37–52, 1987. Citado na página 64.

ZHANG, Q.; LI, H. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *Evolutionary Computation, IEEE Transactions on*, IEEE, v. 11, n. 6, p. 712–731, 2007. Citado na página 17.

ZITZLER, E.; DEB, K.; THIELE, L. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation*, MIT Press, v. 8, n. 2, p. 173–195, 2000. Citado na página 17.

Anexos

ANEXO A – PCA

A análise de componentes principais (PCA) em muitas formas é a base para a análise de dados multivariada. O PCA fornece uma aproximação da tabela de datas, a matriz de dados, X, em termos do produto de duas pequenas matrizes T e P '. Essas matrizes, T e P ', capturam os padrões de dados essenciais de X. Plotar as colunas de T dá uma imagem dos "padrões de objeto"dominantes de X e, analogamente, traçar as linhas de P' mostra os "padrões variáveis"complementares(WOLD; ESBENSEN; GELADI, 1987).

Análise de Componentes Principais (PCA) é uma transformação de base ortogonal. A nova base é encontrada pela diagonalização da matriz de covariância centralizada de um conjunto de dados $\{X_k \in R^N | k = 1, ..., f\}$, definido por $C = ((X_i - (X_k))(X_i - (X_k))^T)$. As coordenadas na base de Autovetores são chamadas de componentes principais. O tamanho de um autovalor>. correspondente a um Autovetores v de C é igual à quantidade de variância na direção de v. Além disso, as direções dos primeiros n Autovetores correspondentes aos maiores n Autovalores cobrem a maior variância possível por n direções ortogonais. Em muitas aplicações, eles contêm as informações mais interessantes: por exemplo, na compactação de dados, em que projetamos nas direções com maior variação para reter o máximo de informações possível ou em desclassificação, onde deliberadamente abandonamos direções com pequena variação(MIKA et al., 1999).